Terascale Optimal PDE Simulations

http://www.tops-scidac.org

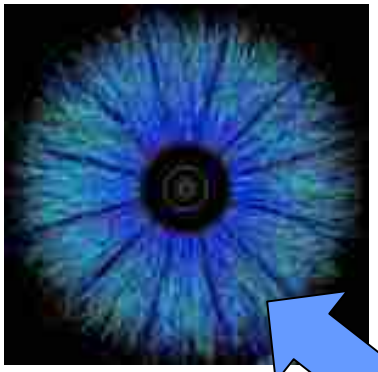
Overview

SciDAC
Scientific Discovery through Advanced Computing

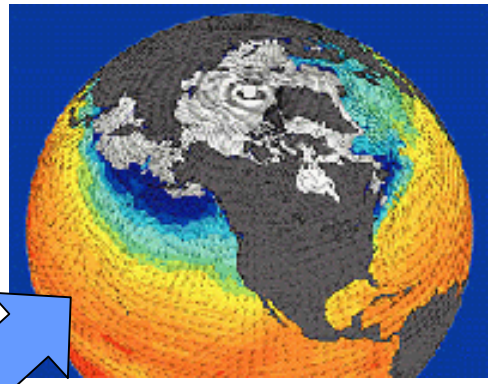
David E. Keyes, Columbia University

"The partial differential equation entered theoretical physics as a handmaid, but has gradually become mistress." – **A. Einstein**

4 projects in high energy and nuclear physics



PDEs

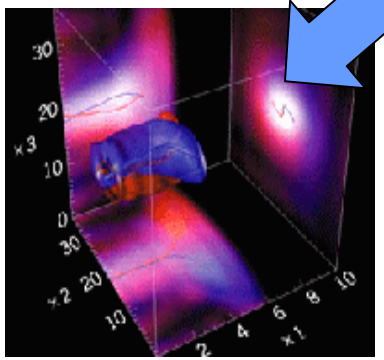


14 projects in biological and environmental research

are dense

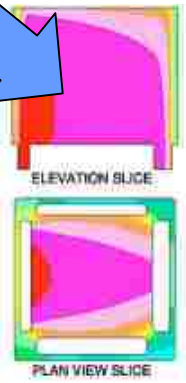
TOPS
Terascale Optimal PDE Simulations
& ISICs

in the SciDAC

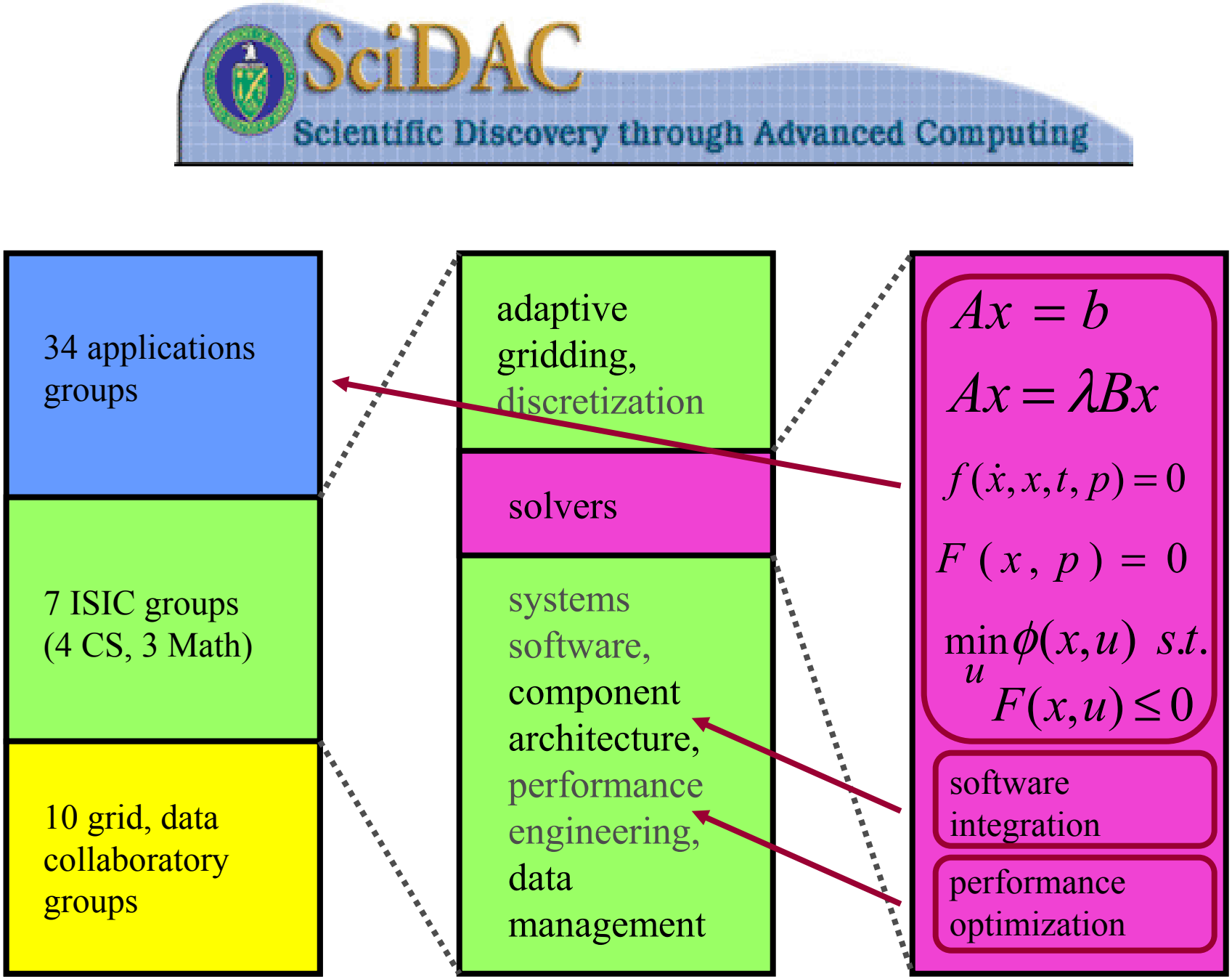


5 projects in fusion energy science

portfolio

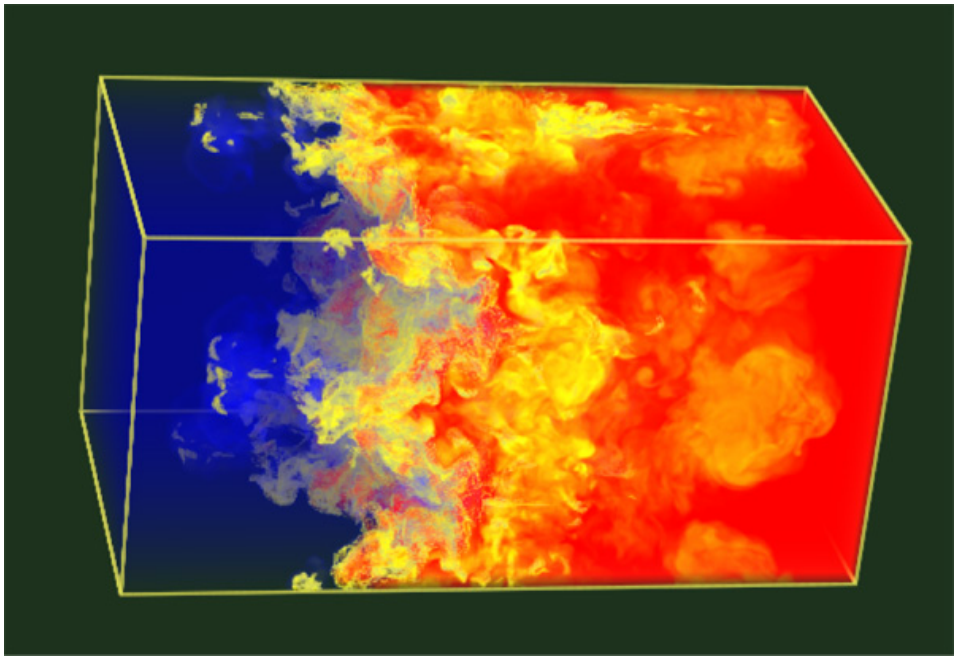


10 projects in basic energy sciences


SciDAC
Scientific Discovery through Advanced Computing
34 applications groups
7 ISIC groups (4 CS, 3 Math)
10 grid, data collaboratory groups
adaptive gridding, discretization
solvers
systems software, component architecture, performance engineering, data management
$Ax = b$
$Ax = \lambda Bx$
$f(\dot{x}, x, t, p) = 0$
$F(x, p) = 0$
$\min_u \phi(x,u) \ \ s.t. \ \ F(x,u) \le 0$
software integration
performance optimization

Imperative: multiple-scale applications

- **Multiple spatial scales**
  - **interfaces, fronts, layers**
  - **thin relative to domain size, $\delta << L$**
- **Multiple temporal scales**
  - **fast waves**
  - **small transit times relative to convection or diffusion, $\tau << T$**



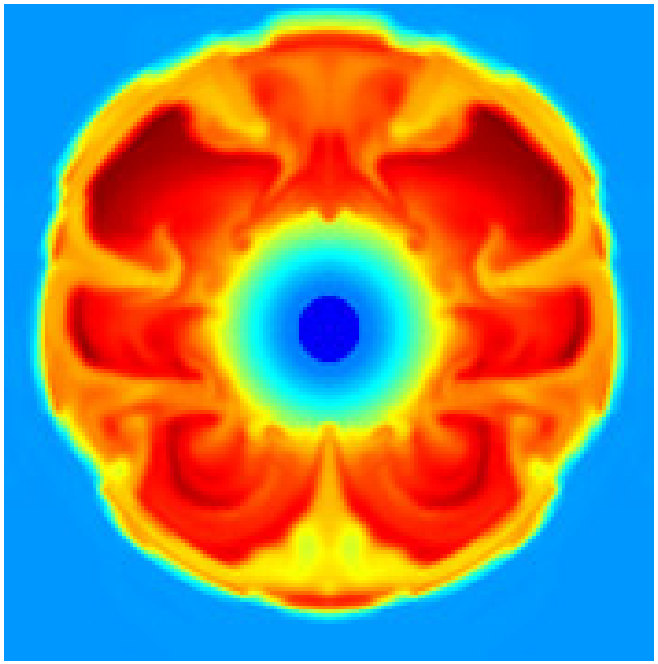
Richtmeyer-Meshkov instability, c/o A. Mirin, LLNL

- **Analyst must *isolate dynamics of interest* and *model the rest* in a system that can be discretized over more modest range of scales (beyond scope of this lecture – application specific)**
- **May lead to local discontinuity or infinitely "stiff" subsystem requiring special treatment by the solution method (our scope in this lecture)**

Examples: multiple-scale applications

- **Biopolymers, nanotechnology**
  - 10^{12} range in time, from 10^{-15} sec (quantum fluctuation) to 10^{-3} sec (molecular folding time)
  - typical computational model ignores smallest scales, works on classical dynamics only, but scientists increasingly want both

- **Galaxy formation**
  - 10^{20} range in space from binary star interactions to diameter of universe
  - heroic computational model handles all scales with localized adaptive meshing



Supernova simulation, c/o A. Mezzacappa, ORNL

- **Supernovae simulation**
  - massive ranges in time and space scales for radiation, turbulent convection, diffusion, chemical reaction, nuclear reaction

Problem characteristics

- Multiple time scales
- Multiple spatial scales
- Linear ill conditioning
- Complex geometry and severe anisotropy
- Coupled physics, with essential nonlinearities
- Ambition for predictability and design from simulation

Multiscale stress on computer architecture

- Spatial resolution stresses memory size
  - number of floating point words
  - precision of floating point words
- Temporal resolution stresses processor clock rate and/or memory bandwidth
- Both stress interprocessor latency, and *together* they *severely* stress memory bandwidth
- Less severely stressed, in principle, are memory latency and interprocessor bandwidth (another talk)
- But "brute force" not an option; need "good" algorithms:
  - Multiscale representations, adaptive meshes, optimal solvers, scalable everything, …

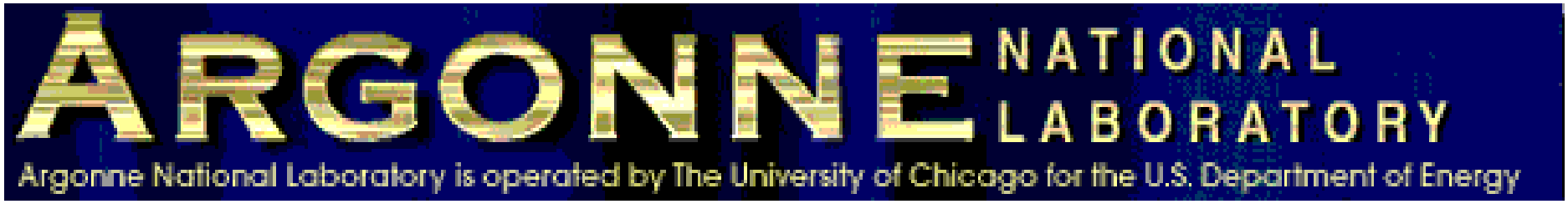
Multiscale stress on algorithms

- **Spatial resolution stresses condition number**
  - **Ill-conditioning: small error in input may lead to large error in output**
  - **For self-adjoint linear systems, cond no.** $\kappa = \| A \| \cdot \| A^{-1} \|$ **— related to ratio of max to min eigenvalue**
  - **For discrete Laplacian,** $\kappa = O(h^{-2})$
  - **With improved resolution we approach the continuum limit of an unbounded inverse**
- **Standard iterative methods fail due to growth in iterations like** $O(\kappa)$ **or** $O(\sqrt{\kappa})$**; we seek** $O(1)$
- **Direct methods fail due to memory growth and bounded concurrency**
- **Solution is *domain decomposed multilevel methods***

Multiscale stress on algorithms, cont.

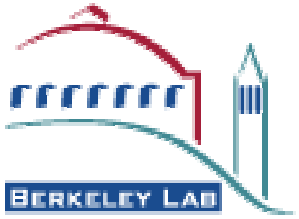
- **Temporal resolution stresses stiffness**
  - **Stiffness: failure to track fastest mode may lead to exponentially growing error in other modes, related to ratio of max to min eigenvalue of A in $y_t = Ay$**
  - **By definition, multiple timescale problems contain phenomena of very different relaxation rates**
  - **Certain idealized systems (e.g., incomp NS) are infinitely stiff**
- **Number of steps to finite simulated time grows, to preserve stability, regardless of accuracy requirements**
- **A solution is to *step over* fast modes by assuming quasi-equilibrium**
- **Throws temporally stiff problems into spatially ill-conditioned regime**

Who we are in TOPS…



… the PETSc and TAO people

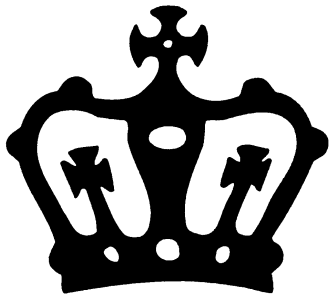
… the Hypre and SUNDIALS people



Berkeley Lab

… the SuperLU and PARPACK people

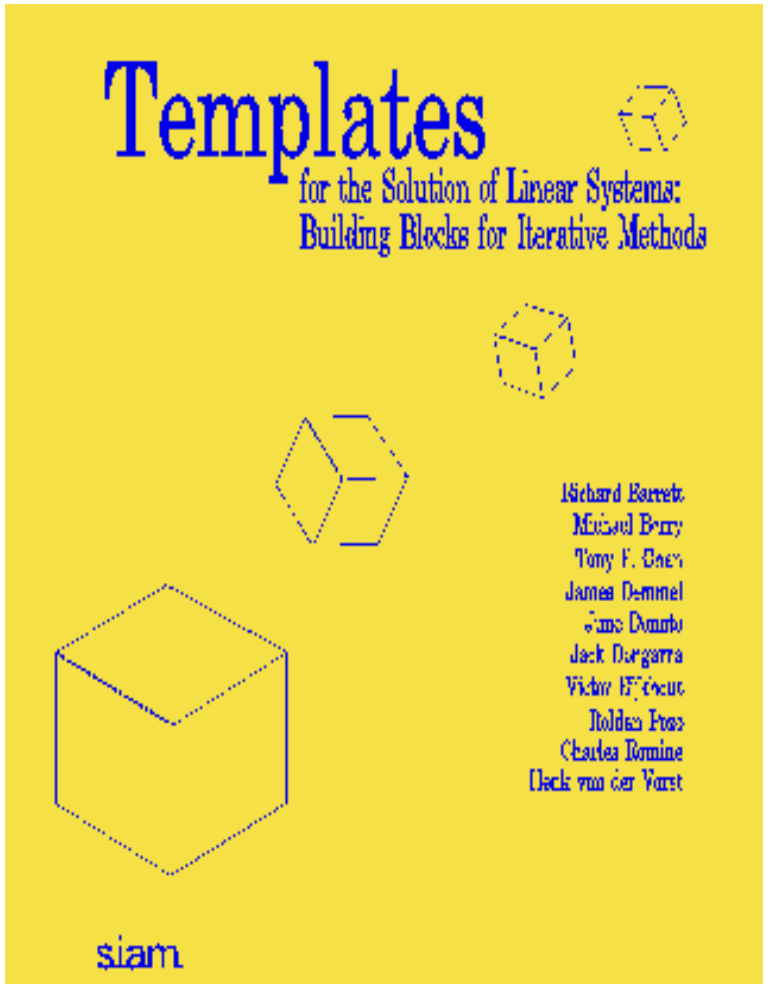
Plus some university collaborators …



… with a history of lab collaborations in high performance computing

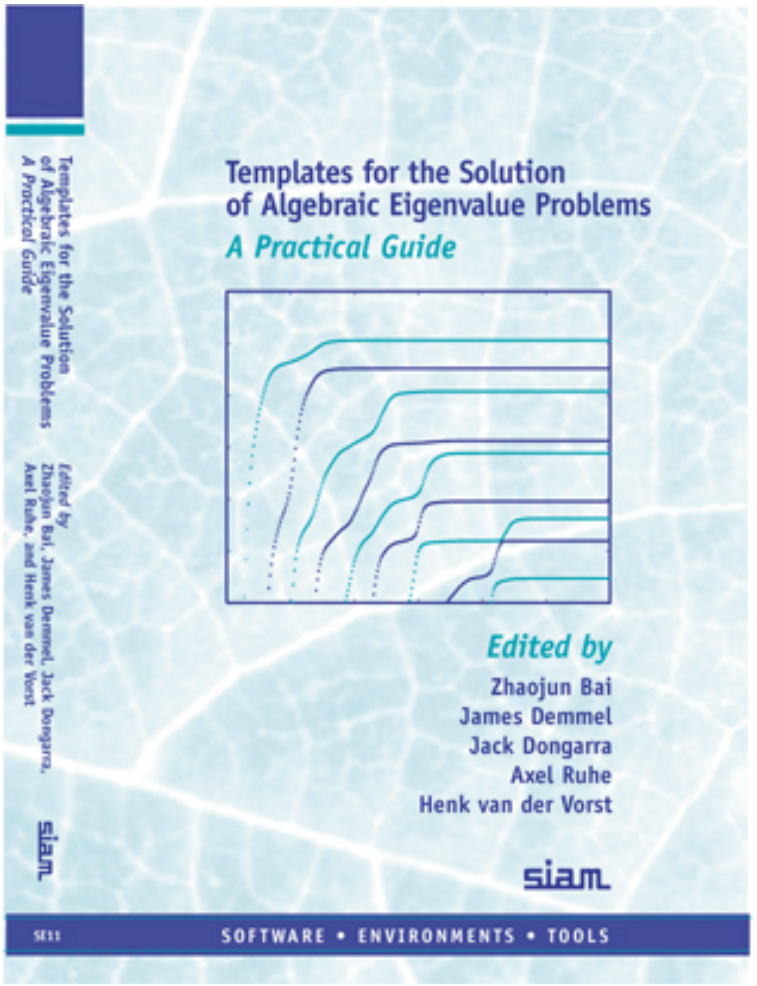
You may know the on-line "Templates" guides …

www.netlib.org/templates



124 pp.

www.netlib.org/etemplates



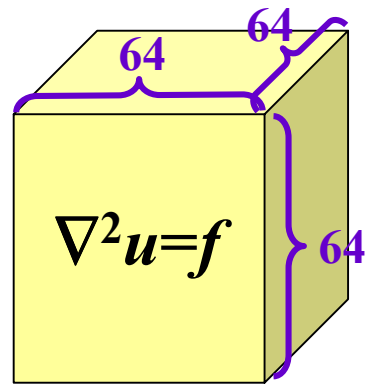
410 pp.

… these are good starts, but not adequate for SciDAC scales!

The power of optimal algorithms

- Advances in algorithmic efficiency can rival advances in hardware architecture
- Consider Poisson's equation on a cube of size $N=n^3$

Year	*Method*	*Reference*	*Storage*	*Flops*
1947	GE (banded)	Von Neumann & Goldstine	n^5	n^7
1950	Optimal SOR	Young	n^3	$n^4 \log n$
1971	CG	Reid	n^3	$n^{3.5} \log n$
1984	Full MG	Brandt	n^3	n^3

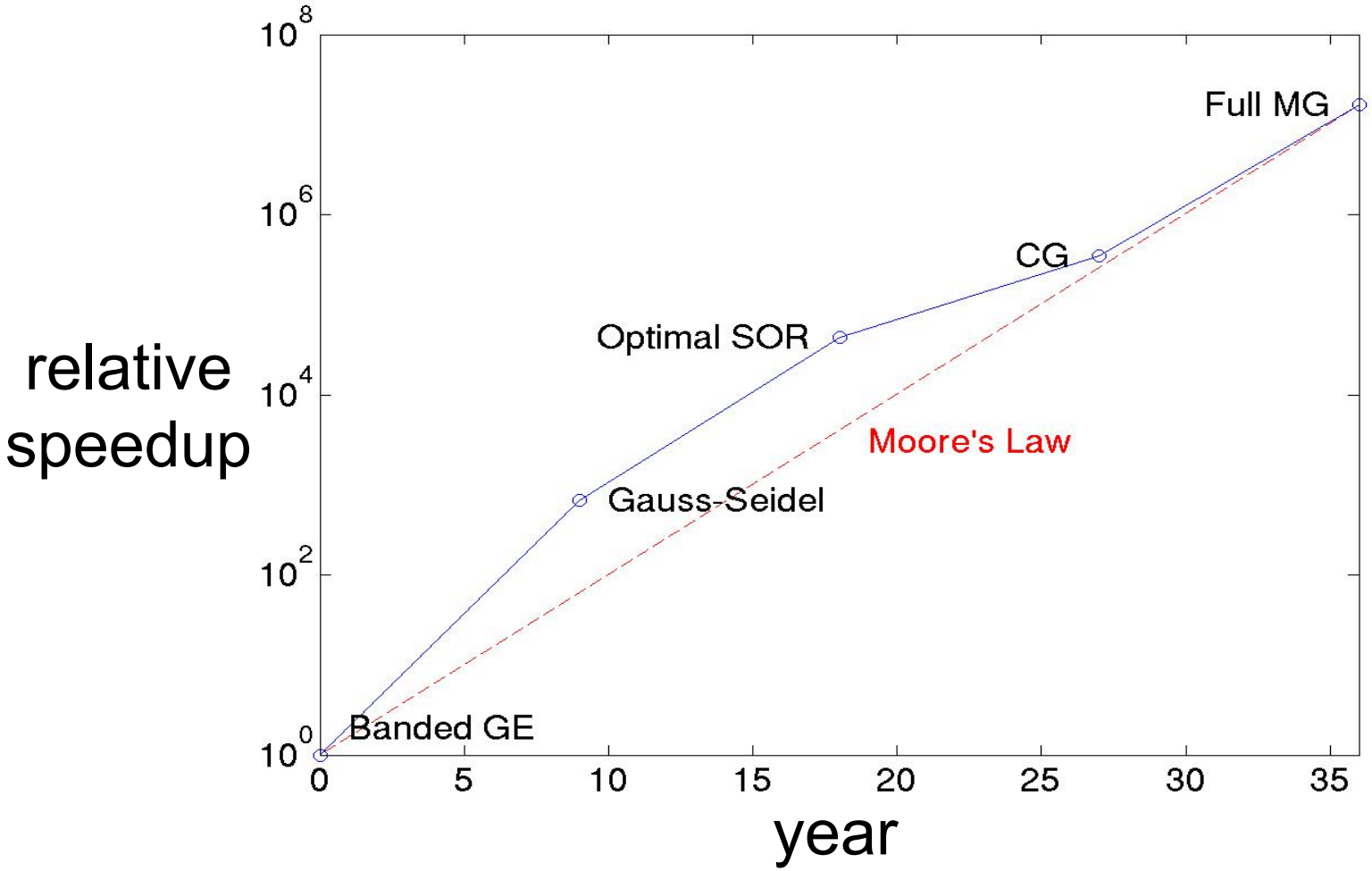


- If $n=64$, this implies an overall reduction in flops of ~16 million*

*Six-months is reduced to 1 s

Algorithms and Moore's Law

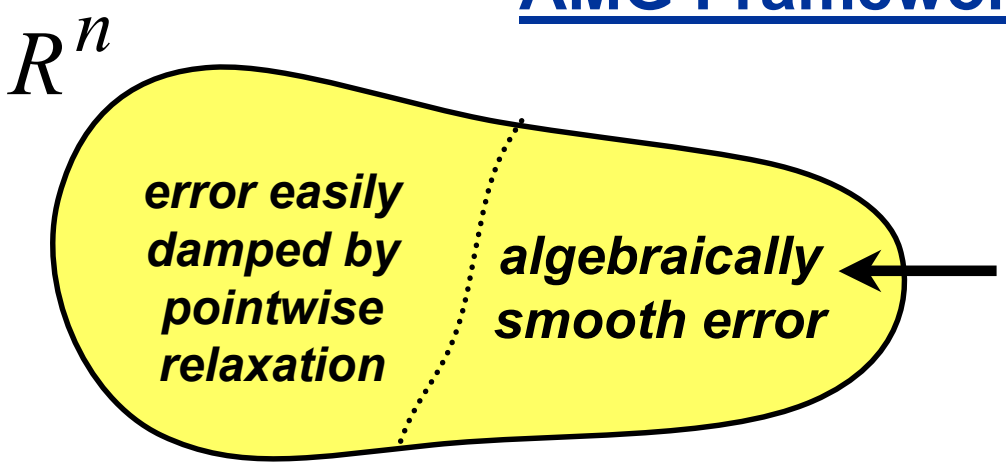
- **This advance took place over a span of about 36 years, or 24 doubling times for Moore's Law**
- **$2^{24} \approx 16$ million $\Rightarrow$ the same as the factor from algorithms alone!**



But where to go past *O(N)* ?

- Since *O(N)* is already optimal, there is nowhere further "upward" to go in efficiency, but one must extend optimality "outward," to more general problems
- Hence, for instance, algebraic multigrid (AMG) to seek to obtain *O(N)* in *indefinite*, *anisotropic*, or *inhomogeneous* problems on *irregular* grids

AMG Framework



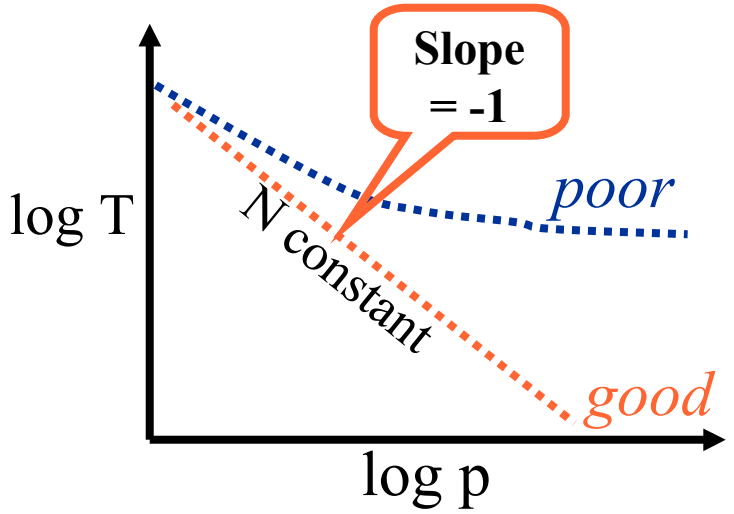
Choose coarse grids, transfer operators, and smoothers to eliminate these "bad" components within a smaller dimensional space, and recur

TOPS is dedicated to the proposition that …

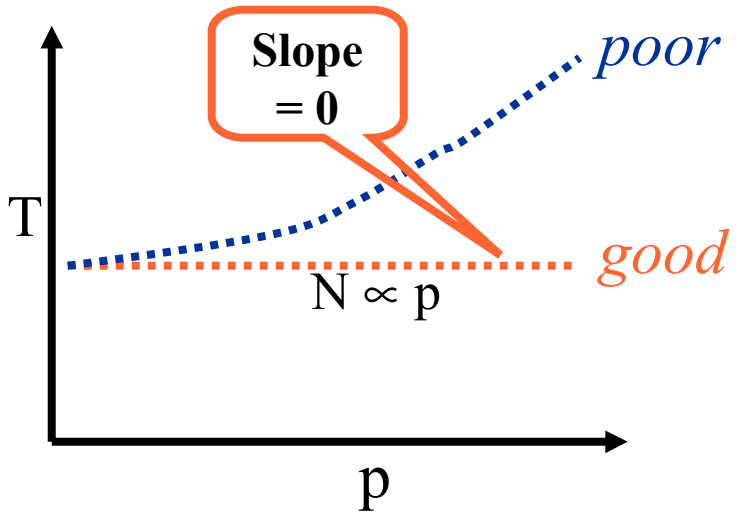
- **Not all problems are created equal**
  - so a large variety of solvers should be callable from one interface
- **Solver software can rarely be thrown over the wall**
  - so we are committed to collaborations with users
- **Discretization and solution rarely separate cleanly**
  - so we are committed to collaborations with ISIC colleagues
- **Desire for resolution will grow without bound**
  - so we concentrate on solvers that scale well (in the "weak" sense)
- **Solving the PDE well is only a beginning, not the end, in doing computational science**
  - so we are providing a software "tool chain" of several links, which are implemented over common data structures and kernel functionality

Two distinct definitions of scalability

- **"Strong scaling"**
  - **execution time decreases in inverse proportion to the number of processors**
  - ***fixed size problem overall***



- **"Weak scaling"**
  - **execution time remains constant, as problem size and processor number are increased in proportion**
  - ***fixed size problem per processor***
  - **also known as "Gustafson scaling"**



TOPS has a dream that users will…

- **Understand range of algorithmic options w/tradeoffs**

  ***e.g.*, memory *vs.* time, comp. *vs.* comm., inner iteration work *vs.* outer**

- **Try all reasonable options "easily"**

  without recoding or extensive recompilation

- **Know how their solvers are performing**

  with access to detailed profiling information

- **Intelligently drive solver research**

  ***e.g.*, publish joint papers with algorithm researchers**

- **Simulate *truly new physics* free from solver limits**

  ***e.g.*, finer meshes, complex coupling, full nonlinearity**

A project like TOPS can be useful because …

- **Many applications are presently solver-bound**

  ***e.g.*, 90-95% of execution time in solver, limited to 1 or 2 dimensions**

- **Many apps ambitions are too low, focused too near**

  concentrated on getting a few big runs, without enough validation and verification, since iteration over the "forward" problem is costly

- **Community codes are hard to keep current**

  slow process to implement new algorithms, to port to new machines

- **Computer scientists need good stepping stone to apps**

  solvers are good target for research in components and performance

- **Code developers need good solvers, too**

  from scalable Poisson solves to mesh optimization, from sea to shining sea …

TOPS set out with certifiably good ingredients

- **Constituent software powers commercial toolkits**
  - ***e.g.*, SUNDIALS (*Mathematica*), SuperLU (*Matlab*), PETSc (numerous)**
- **Constituent software powers major research codes**
  - ***e.g.*, Hypre (ASCI), PETSc (NASA HPC, Harvard Med)**
- **Constituent software has powered prizes**
  - ***e.g.*, PETSc (Bell Prize), Veltisto ("Best Paper" at SC)**
- **… and science on covers of *Science* and *Nature***
  - ***e.g.*, SuperLU, ScaLAPACK**
- **TOPS ingredients are continually being improved, in conjunction with thousands of computational scientists and engineers *around the world***

What value is being added by TOPS today?

- **Interoperability for new performance**

  ***e.g.*, Hypre preconditioners in PETSc**

- **Interoperability for new functionality**

  ***e.g.*, PETSc in TAO and Veltisto for large-scale optimization**

- **Interoperability for new CS research**

  ***e.g.*, componentization of PETSc and Hypre**

- **Development and maintenance of core codes**
- **Expansion of user consulting capability**
- **Education and training of next generation of solver developers**
- **Outreach to applications community**

Outline for presentation

- Introduction & motivation (just completed ☺)
- TOPS scientific overview (broad and shallow)
  - algorithmic research and development (5 areas)
  - infrastructural research and development (2 areas)
  - applications collaborations (3 major groups; 7 others)
- TOPS philosophy

break

- Nonlinear robustification techniques
- Linear preconditioning techniques for nonlinear problems
  - standard
  - physics-based
  - applications to PDE-constrained optimization

Scope for TOPS

- **Design and implementation of "solvers"**
  - **Linear solvers**

$$Ax = b$$

  - **Eigensolvers**

$$Ax = \lambda Bx$$

  - **Nonlinear solvers (w/ sens. anal.)**

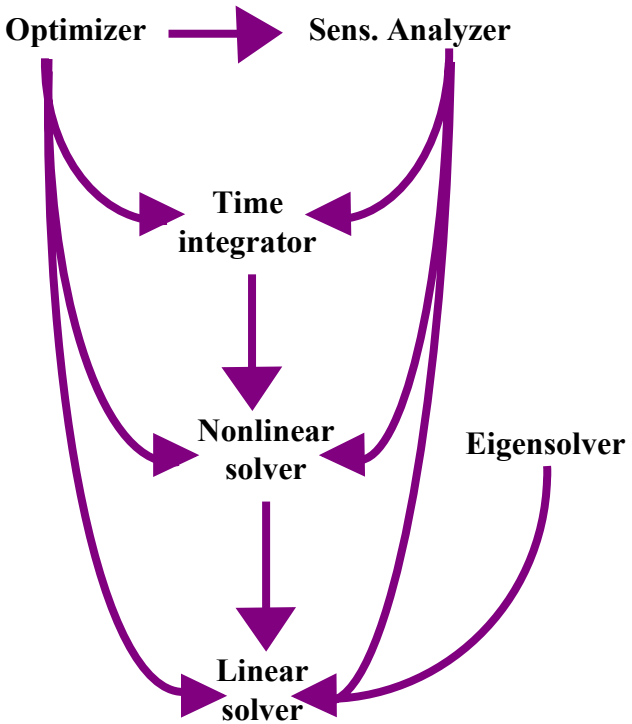
$$F(x, p) = 0$$

  - **Time integrators (w/ sens. anal.)**

$$f(\dot{x}, x, t, p) = 0$$

  - **Optimizers**

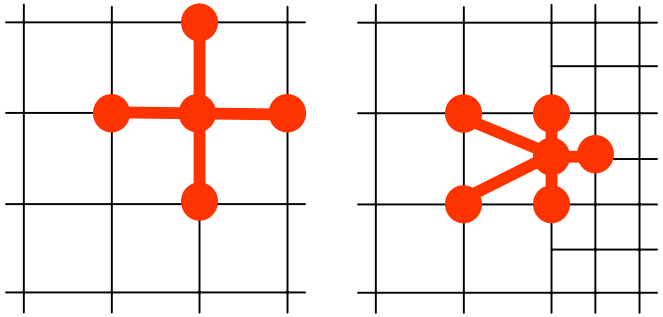
$$\min_u \phi(x,u) \; s.t. \; F(x,u) = 0, u \geq 0$$



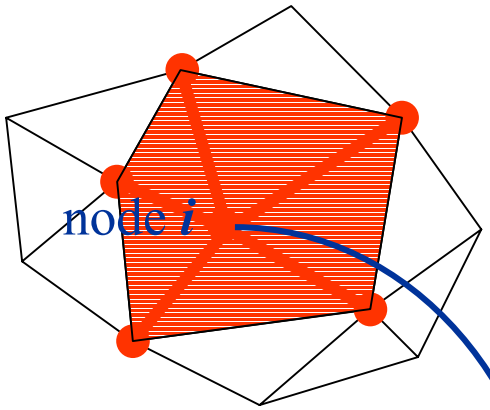
- **Software integration**
- **Performance optimization**

Dominant data structures are grid-based

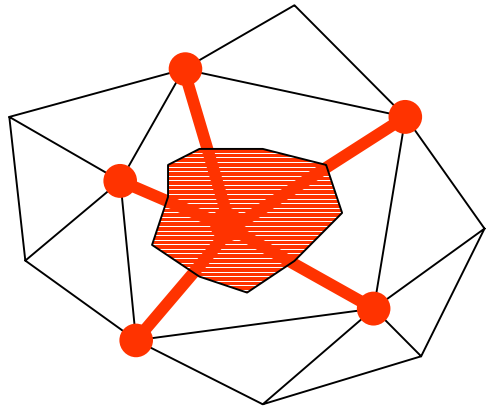
finite differences



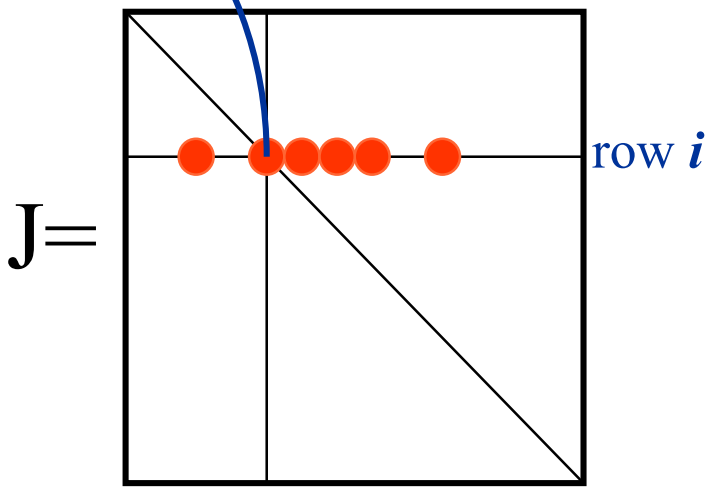
finite elements



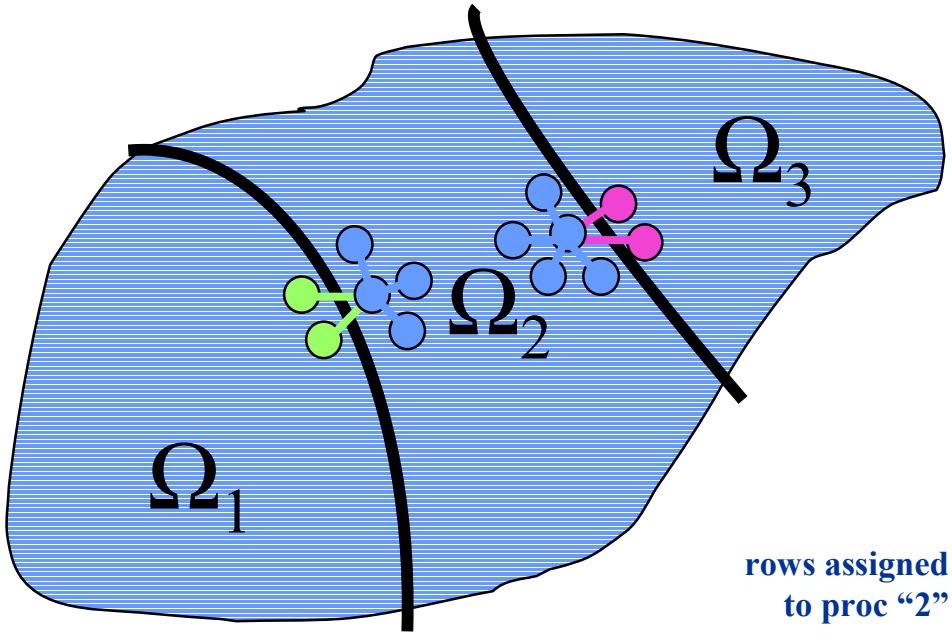
finite volumes



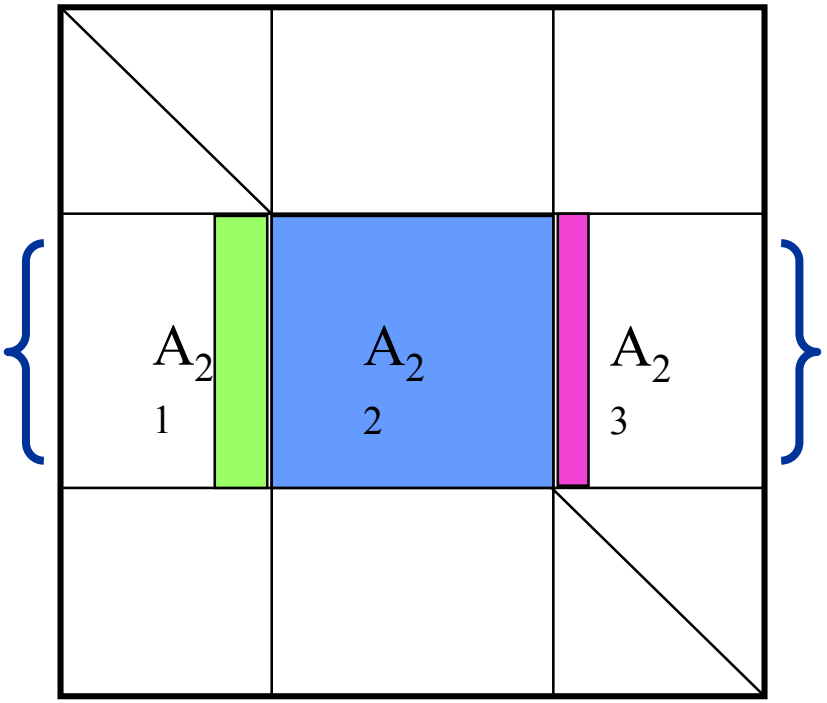
All lead to problems with sparse Jacobian matrices



SPMD parallelism w/domain decomposition



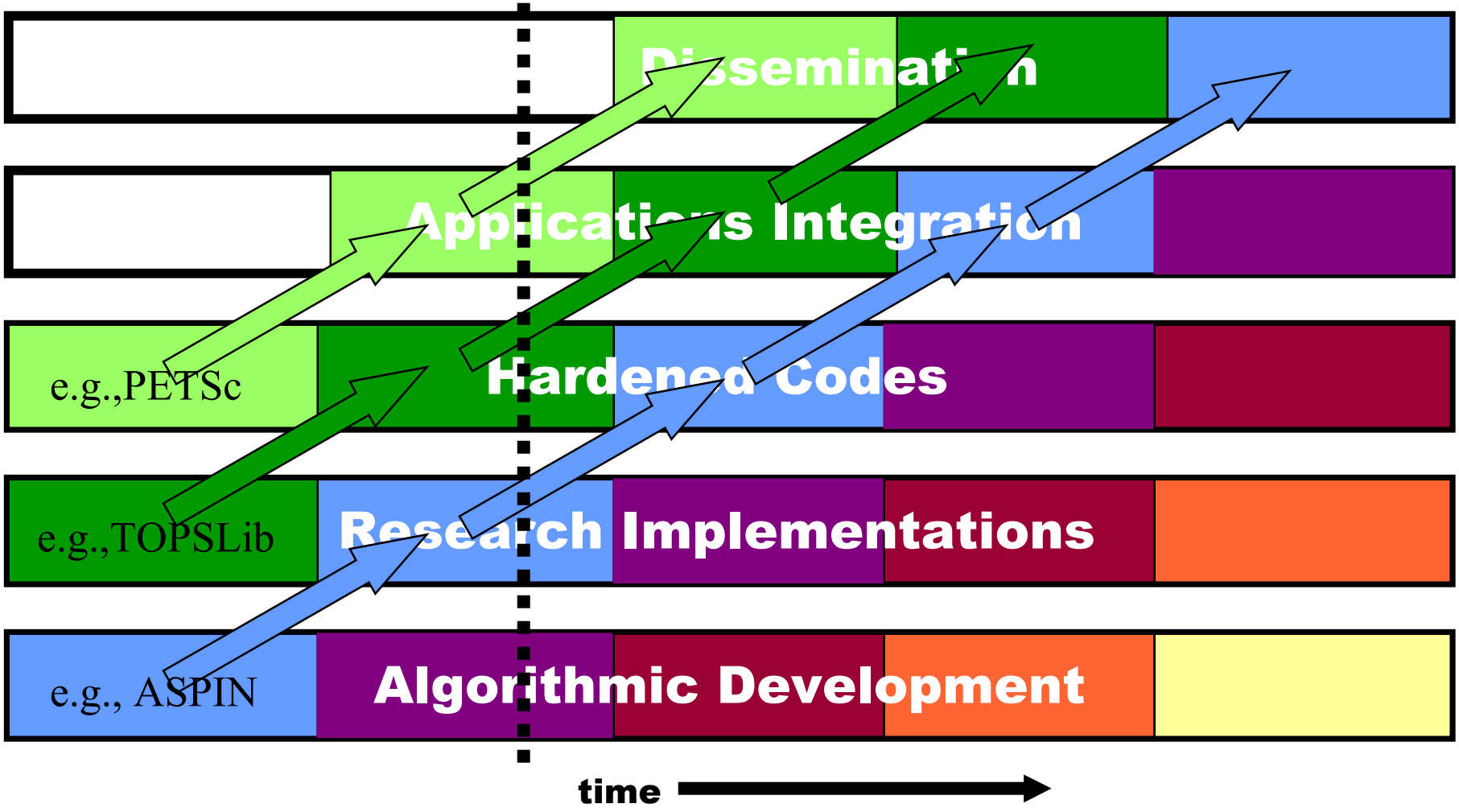
rows assigned
to proc "2"



Partitioning of the grid induces block structure on the Jacobian

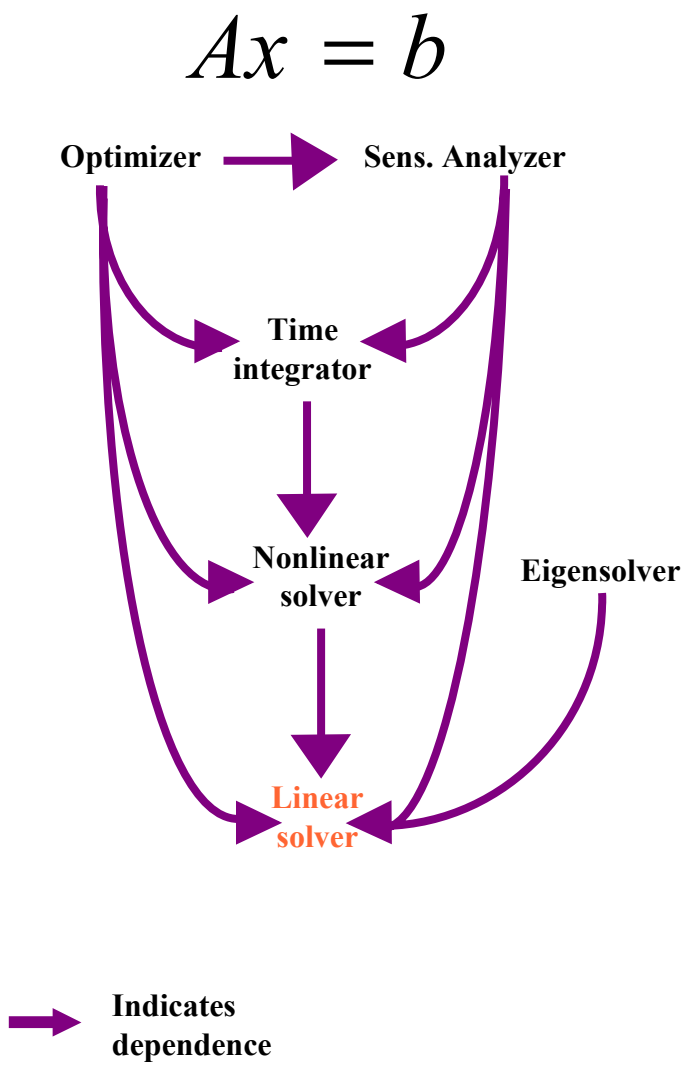
Abstract Gantt Chart for TOPS

Each color module represents an algorithmic research idea on its way to becoming part of a supported community software tool. At any moment (vertical time slice), TOPS has work underway at multiple levels. While some codes are in applications already, they are being improved in functionality and performance as part of the TOPS research agenda.



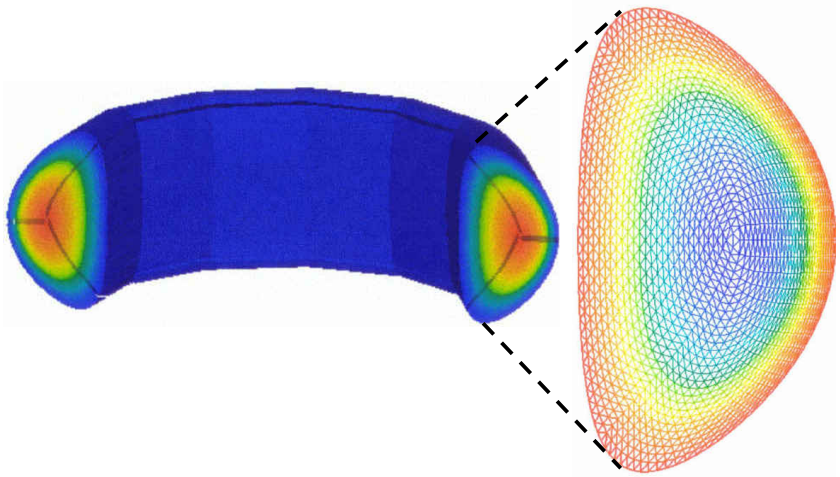
Linear solvers

- One of the greatest investments in the history of DOE numerical computing and the largest and core part of TOPS
- TOPS features the workhorse combo of Krylov preconditioned with algebraic multigrid, geometric multigrid, various incomplete factorizations, as well as direct methods – all sparse oriented
- Also research on innovative methods like adaptive AMG, FOSLS, hierarchical ILU, and adaptive multi-method solvers
- Extensive research on scalability features and memory-adaptive versions of direct methods



Linear solvers progress

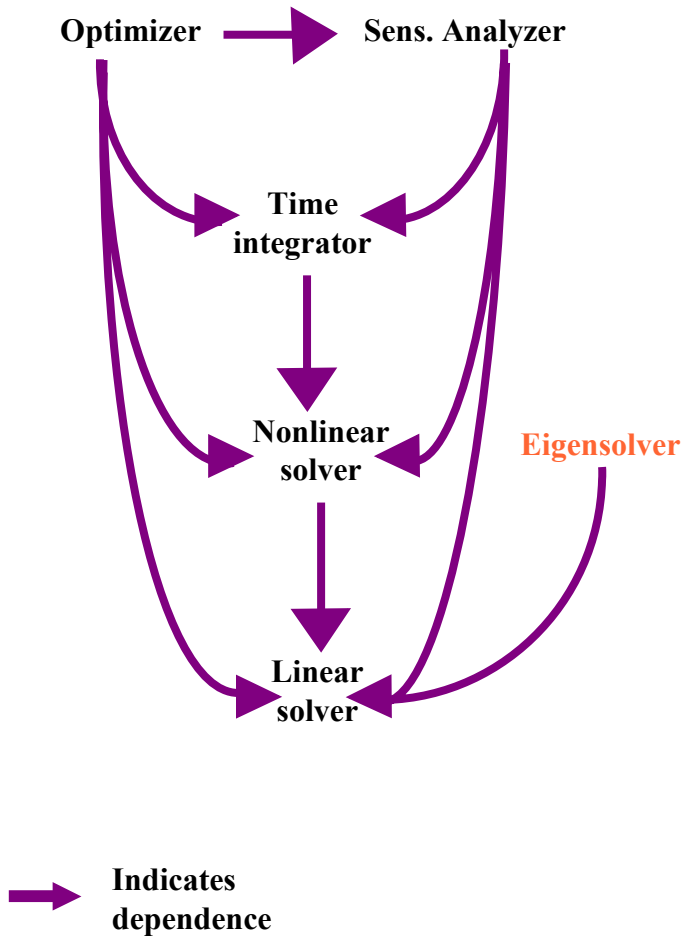
- Algebraic multigrid is dependent upon heuristics to make up for geometric information and to extend optimal convergence from the elliptic regime (where geometric and algebraic smoothness are the same) to more general problems
- When applying AMG anew, must occasionally extend the set of heuristics, sometimes using information beyond matrix alone; self-adaptive AMG a new holy grail
- On software side, also extending the set of interfaces to get closer to user data structures
- MG needs coarse solves, which is one reason for on-going direct methods research
- Sometimes no alternative to direct methods, including in shift-invert



Eigensolvers

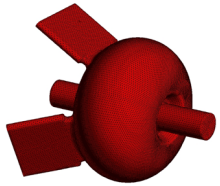
- Preferred eigenanalysis algorithm depends upon: system structure, computational resources, and portion of spectrum and invariant subspaces desired
- Based on customer, TOPS currently concentrates on sparse, symmetric, and small subrange of high-dimensional spectrum
- Exact-shift-invert Lanczos and Jacobi-Davidson both important, preference depending in part on memory available
- Innovative research also in multilevel eigensolvers and in sparse QR

$$Ax = \lambda Bx$$

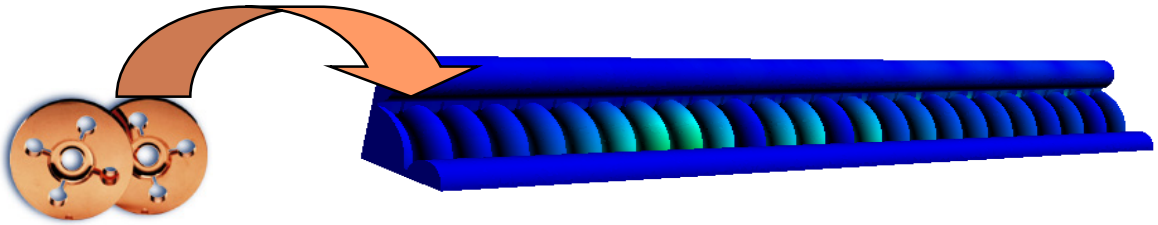


Eigensolvers progress

- AST's **Omega3P** is using TOPS software to find EM modes of accelerator cavities, currently lossless (lossy to come)



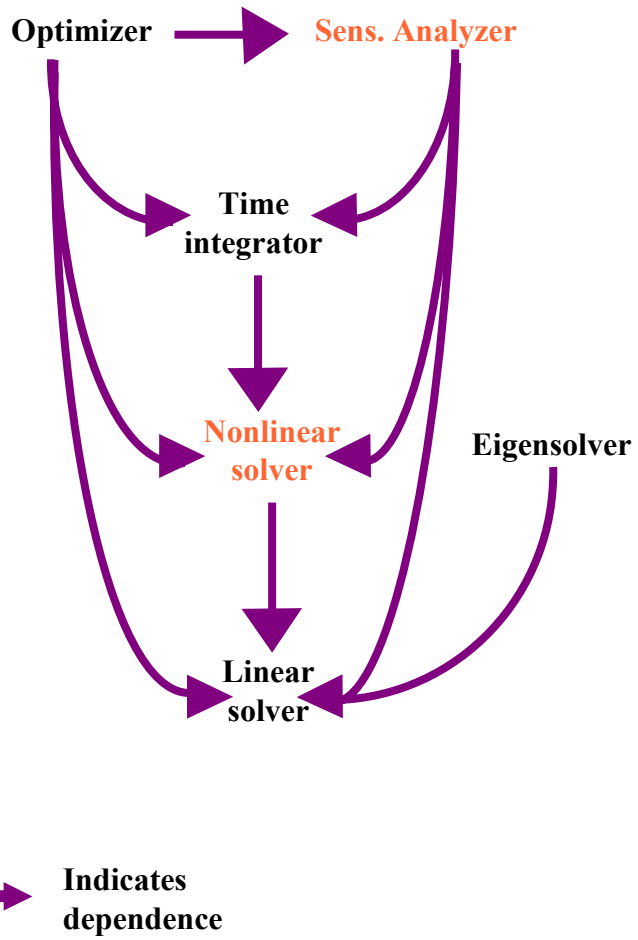
- Methods: Exact Shift-and-Invert Lanczos (ESIL), combining **PARPACK** with **SuperLU** when there is sufficient memory, and Jacobi-Davidson otherwise
- Current high-water marks:
  - 47-cell chamber, finite element discr. of Maxwell's eqs.
  - System dimension 1.3 million
  - 20 million nonzeros in system, 350 million in LU factors
  - *halved analysis time* on 48 processors, scalable to many hundreds



Nonlinear solvers

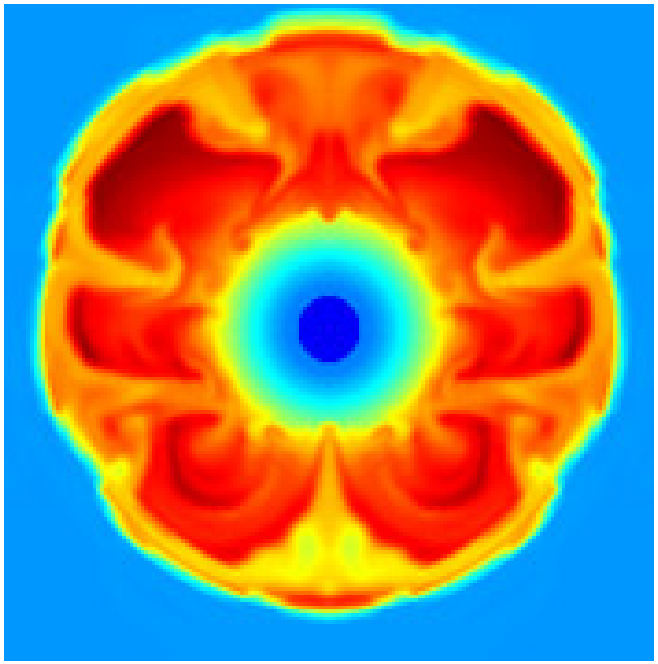
- TOPS features two "workhorse" methods, Newton-Krylov-Schwarz and Newton-Krylov-multigrid, plus two methods in research stages, nonlinear Schwarz (ASPIN) and nonlinear multigrid (FAS)
- Newton implies the ability to solve linear systems with the Jacobian, which leads instantly to sensitivity and optimization capabilities rarely present in legacy codes
- "Jacobian-free" versions of NKS and NK-MG do not require users to supply Jacobian evaluation routines
- Also researching nonlinear versions of substructuring DD methods

$$F(x, p) = 0$$



Nonlinear solver progress

- Mature algorithmic technology in mature software design, with hooks for user-supplied "physics-based" preconditioning
- Newton robustification required
- Pseudo-transient continuation, mesh sequencing, and mainstream algebraic techniques (linesearch and trustregion) available in **PETSc** and **SUNDIALS**



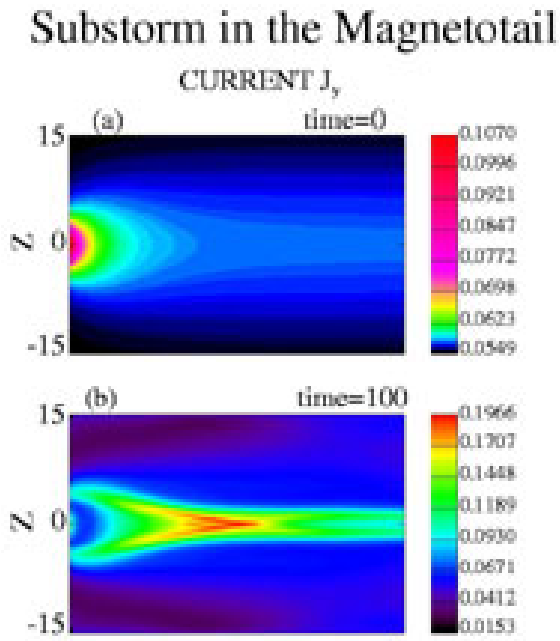
- Difficult "sell" to get users to embrace Newton after lifetime of splitting and linearization
- Built demo of a Hall MHD computation directly into **PETSc** release

Next: 6-slide interlude on NK-MG

Ex.: nonlinear solvers in Hall MR

Magnetic Reconnection: Applications to Sawtooth Oscillations, Error Field Induced Islands and the Dynamo Effect

The research goals of this project include producing a unique high performance code and using this code to study magnetic reconnection in astrophysical plasmas, in smaller scale laboratory experiments, and in fusion devices. The modular code that will be developed will be a fully three-dimensional, compressible Hall MHD code with options to run in slab, cylindrical and toroidal geometry and flexible enough to allow change in algorithms as needed. The code will use adaptive grid refinement, will run on massively parallel computers, and will be portable and scalable. The research goals include studies that will provide increased understanding of sawtooth oscillations in tokamaks, magnetotail substorms, error-fields in tokamaks, reverse field pinch dynamos, astrophysical dynamos, and laboratory reconnection experiments.



PI: Amitava Bhattacharjee
University of Iowa

Status of CMRS collaboration

- CMRS team has provided TOPS with discretization of model 2D multicomponent MHD evolution code in PETSc's DMMG format using automatic differentiation for Jacobian objects
- TOPS has implemented fully nonlinearly implicit GMRES-MG-ILU parallel solver with deflation of nullspace in CMRS's doubly periodic formulation
- CMRS and TOPS reproduce the same dynamics on the same grids with the same time-stepping, up to a finite-time singularity due to collapse of current sheet (that falls below presently uniform mesh resolution)
- TOPS code, being implicit, can choose timesteps an order of magnitude larger, with potential for higher ratio in more physically realistic parameter regimes, but is presently slower in wall-clock time
- Plan: tune PETSc solver by profiling, blocking, reuse, etc.
- Plan: go higher-order in time
- Plan: identify the numerical complexity benefits from implicitness (in suppressing fast timescales) and quantify (explicit versus implicit)
- Plan (with APDEC team): incorporate AMR

2D Hall MHD sawtooth instability

(Porcelli *et al.*, 1993, 1999)

Model equations:

$$\frac{\partial F}{\partial t} + [\phi, F] = \rho_s^2 [U, \psi]$$

$$\frac{\partial U}{\partial t} + [\phi, U] = [J, \psi]$$

$$F = \psi + d_e^2 J$$

$$J = -\nabla^2 \psi$$

$$U = \nabla^2 \phi$$

ex29.c in PETSc 2.5.1

$$\text{with} [A, B] = \hat{z} \cdot \nabla A \times \nabla B$$

$$\vec{B} = B_0 \hat{z} + \nabla \psi \times \hat{z}$$

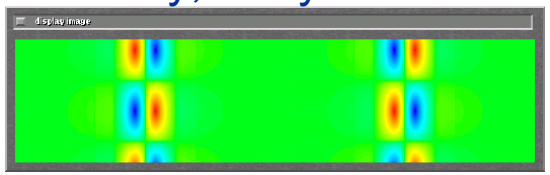
$$\vec{v} = \hat{z} \times \nabla \phi$$

Equilibrium:

$$\phi_{eq} = U_{eq} = 0$$

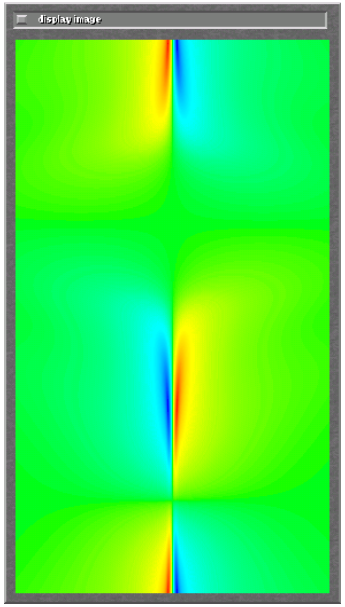
$$\psi_{eq} = J_{eq} = \cos x \quad , \quad F_{eq} = (1 + d_e^2) \cos x$$

Vorticity, early time



Vorticity, later time

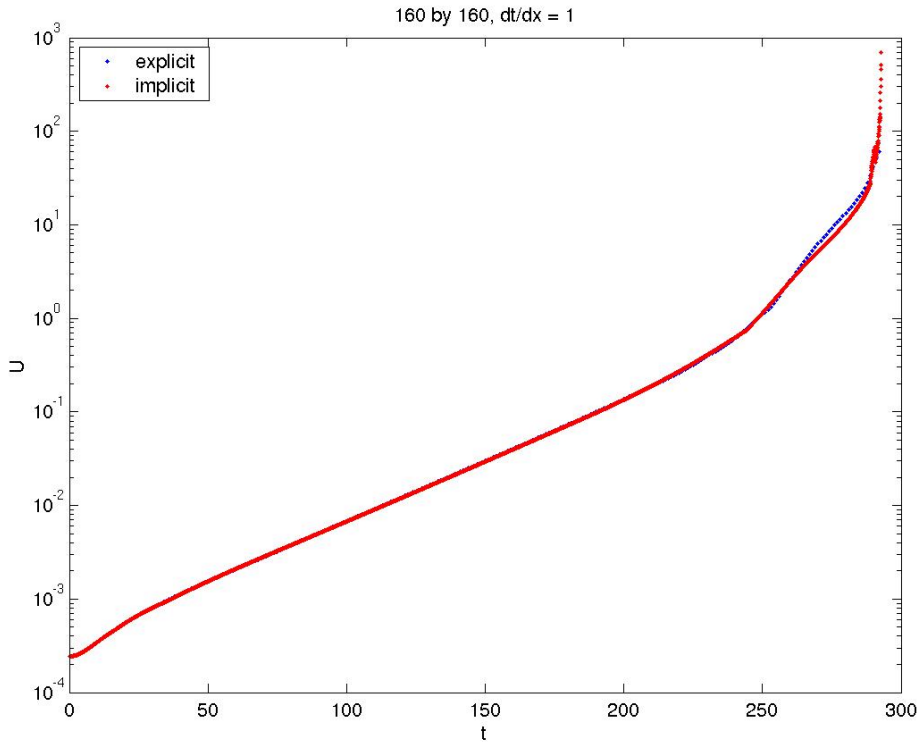
zoom



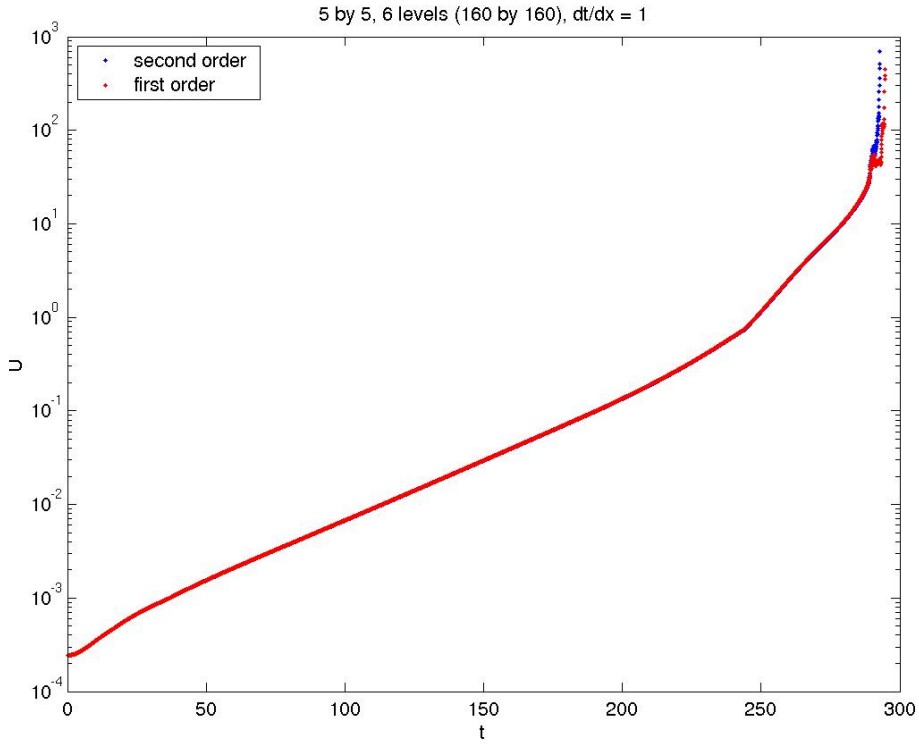
figures c/o A. Bhattacharjee, CMRS

PETSc's DMMG in Hall MR application

- **Implicit code (snes/ex29.c) versus explicit code (sles/ex31.c), both with second-order integration in time**

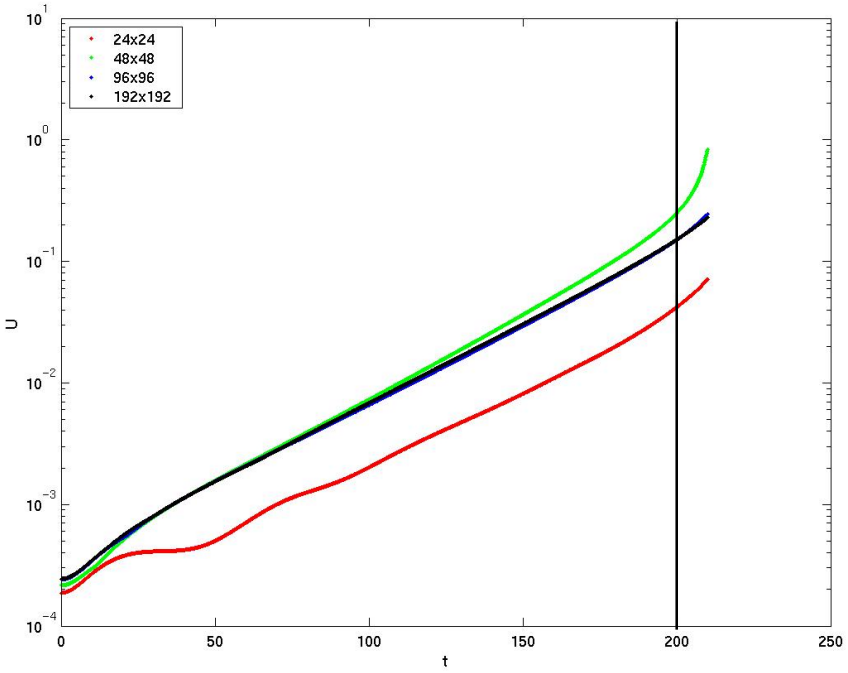


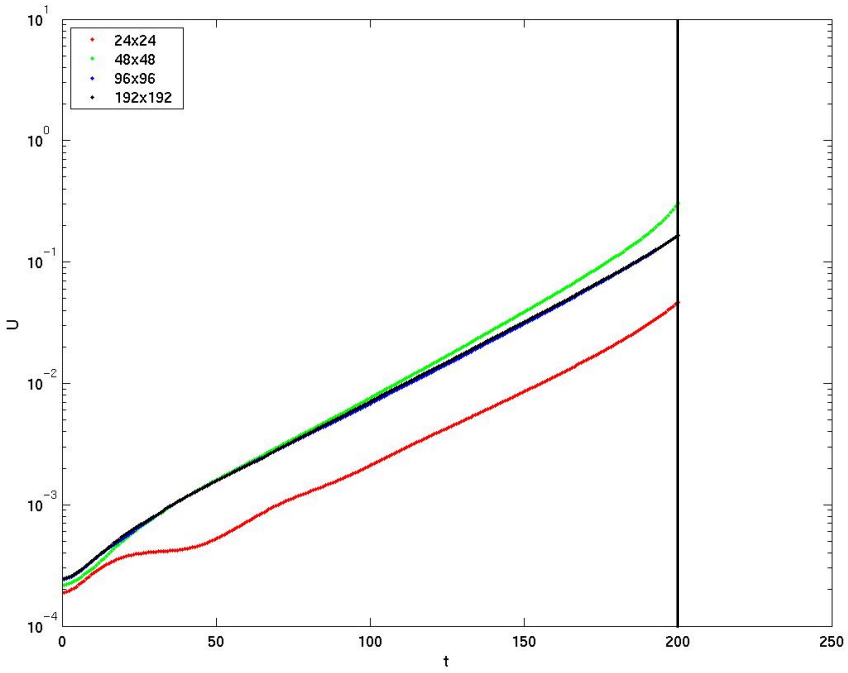
- **Implicit code (snes/ex29.c) with first- and second-order integration in time**



PETSc's DMMG in Hall MR application

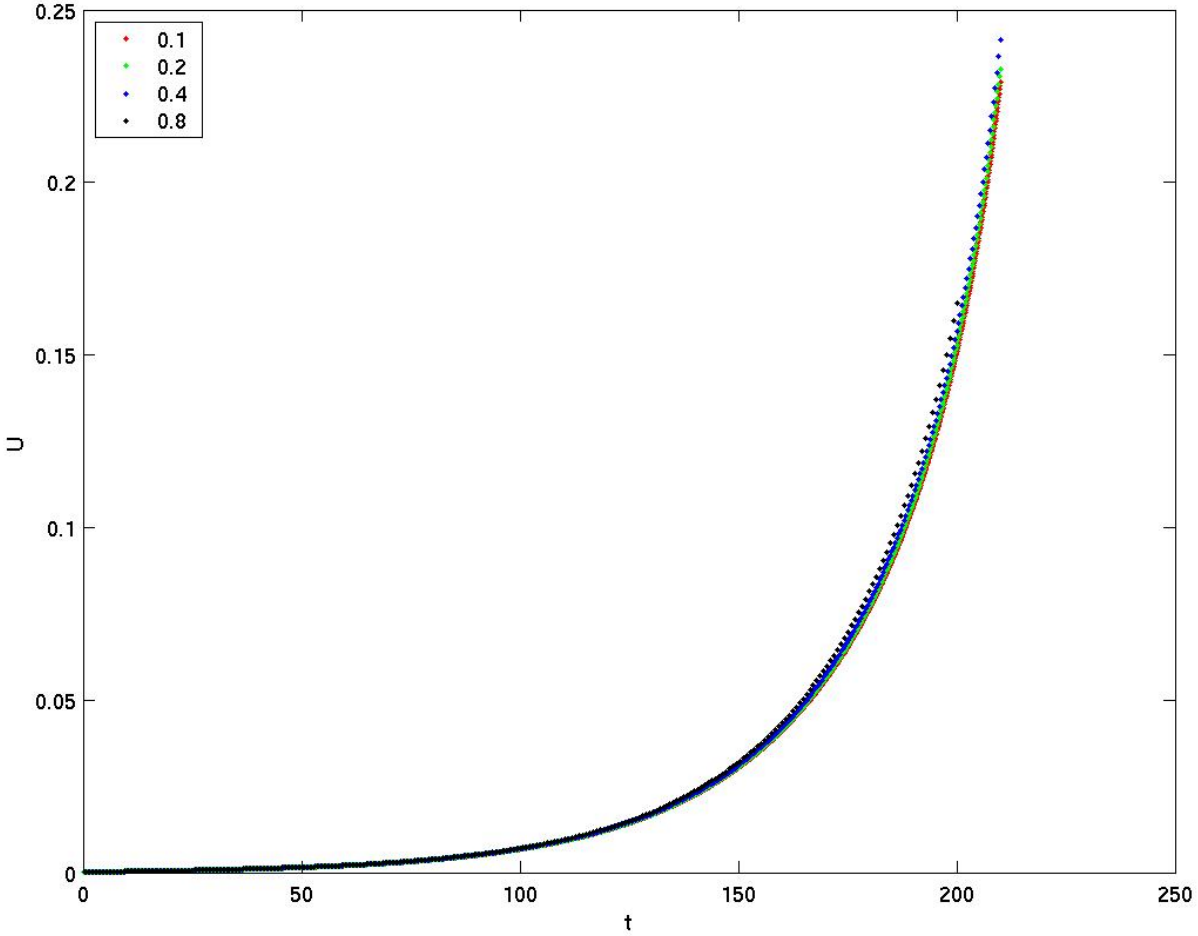
- Mesh and time refinement studies of CMRS Hall magnetic reconnection model problem (4 mesh sizes, dt=0.1 (nondimensional, near CFL limit for fastest wave) on left, dt=0.8 on right)
- Measure of functional inverse to thickness of current sheet versus time, for $0<t<200$ (nondimensional), where singularity occurs around $t=215$





PETSc's DMMG in Hall MR app., cont.

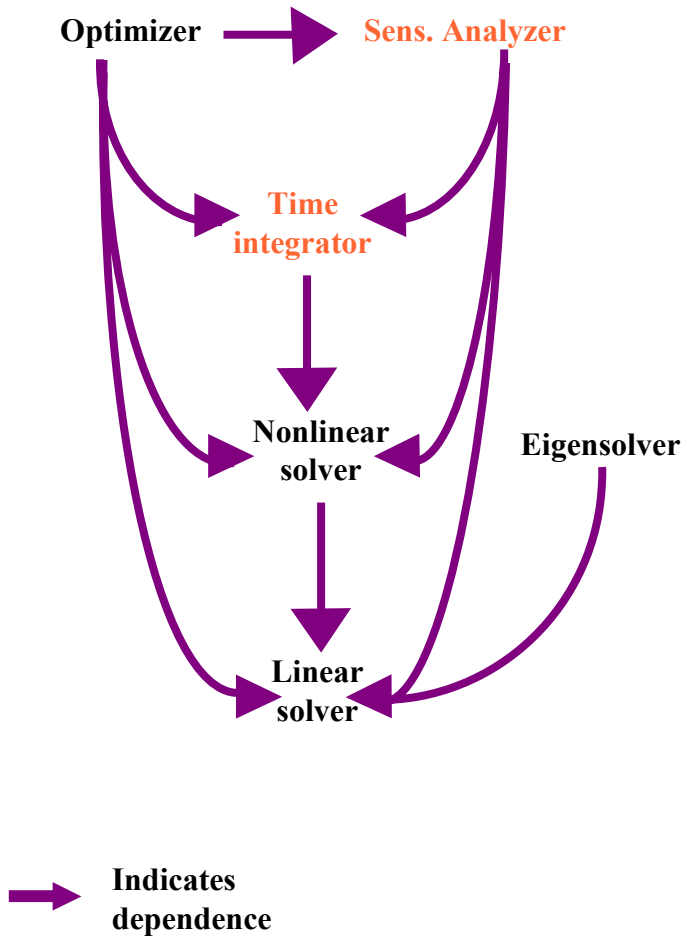
- **Implicit timestep increase studies of CMRS Hall magnetic reconnection model problem, on finest (192×192) mesh of previous slide, in absolute magnitude, rather than semi-log**



Time integrators w/ sensitivity analysis

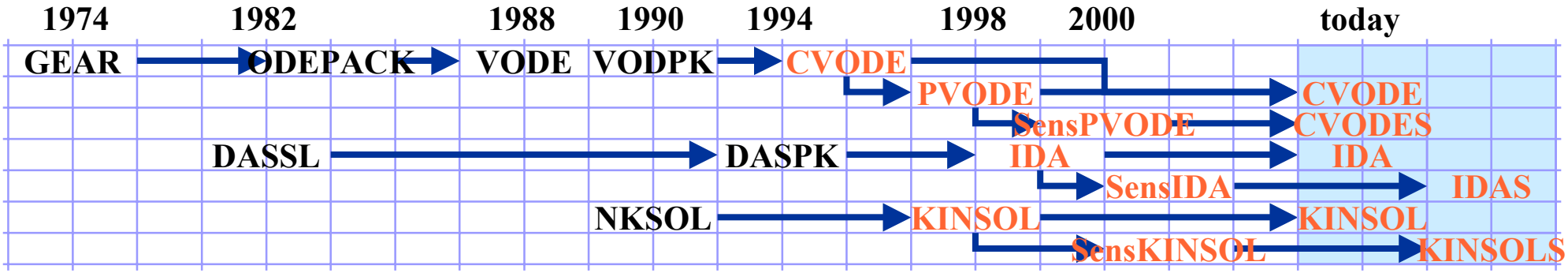
- Transient multirate problems require stiff integrators, a known art, assuming a powerful nonlinear solver capability
- SUNDIALS and PETSc both implement the PVODE backward differentiation schemes for temporal discretization
- PETSc supplies a variety of distributed data structures
- Users who want to use their own data structures, or to utilize built-in sensitivity estimation may prefer SUNDIALS
- Especially recommended for parameterized applications, requiring uncertainty quantification

$$f(\dot{x}, x, t, p) = 0$$



Integrators progress

- **PVODE, IDA, and KINSOL (an NK solver) now wrapped together in SUNDIALS and augmented with forward and adjoint sensitivity analysis capabilities**
- **Embodies decades of work in variable-order, variable-timestep method-of-lines and Newton-Krylov solvers at LLNL**



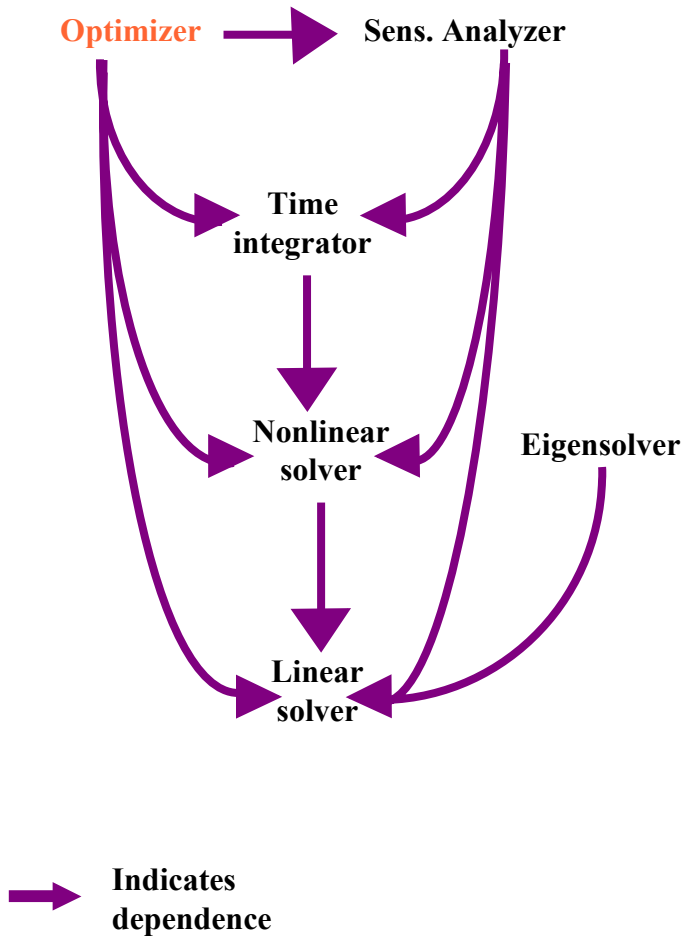
SUNDIALS

SUite of Nonlinear and DIfferential/ALgebraic equation Solvers

Optimizers

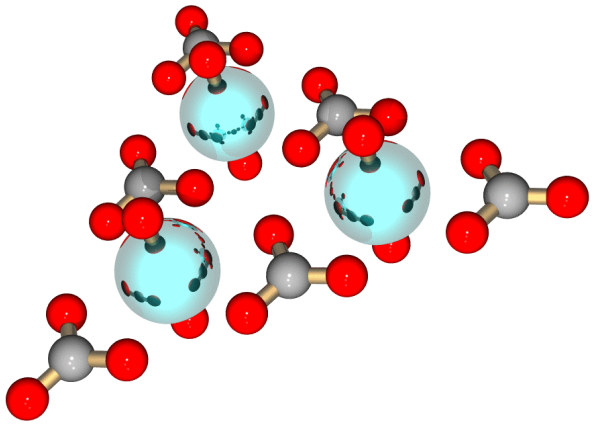
$$\min_u \phi(x,u)\ s.t.\ F(x,u)=0, u\geq 0$$

- Many SciDAC simulations are properly posed as optimization problems, but this may not always be recognized
- Unconstrained or bound-contrained applications use **TAO**
- PDE-constrained problems use **Veltisto**
- Both are built on **PETSc** solvers (and **Hypre** preconditioners)
- **TAO** makes heavy use of AD, freeing user from much coding
- **Veltisto**, based on RSQP, switches as soon as possible to an "all-at-once" method and minimizes the number of PDE solution "work units"



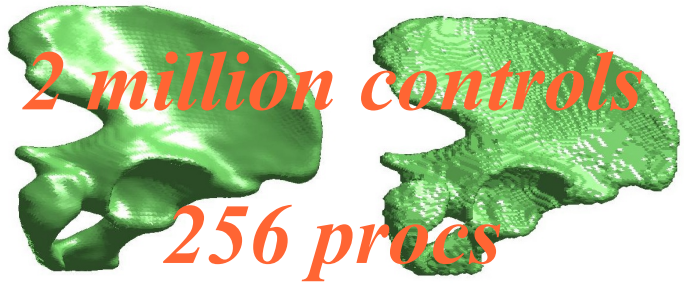
Optimizers progress

- **Unconstrained or bound-constrained optimization**
  - **TAO (interfaced in CCTTSS component framework) used in quantum chemistry energy minimization**



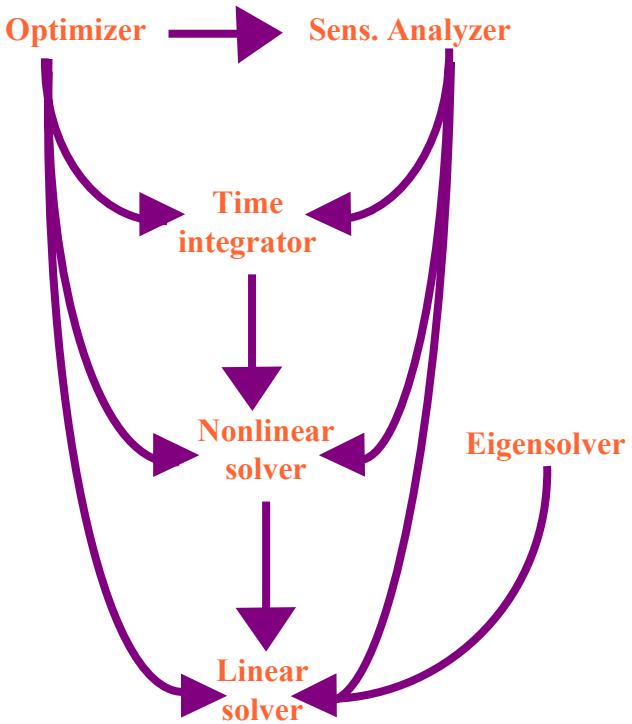
- **PDE-constrained optimization**
  - **Veltisto used in flow control application, to straighten out wingtip vortex by wing surface blowing and sunction; performs full optimization in the time of just five N-S solves**

- **"Best technical paper" at SC2002 went to TOPS team**
  - **Inverse wave propagation employed to infer hidden geometry**



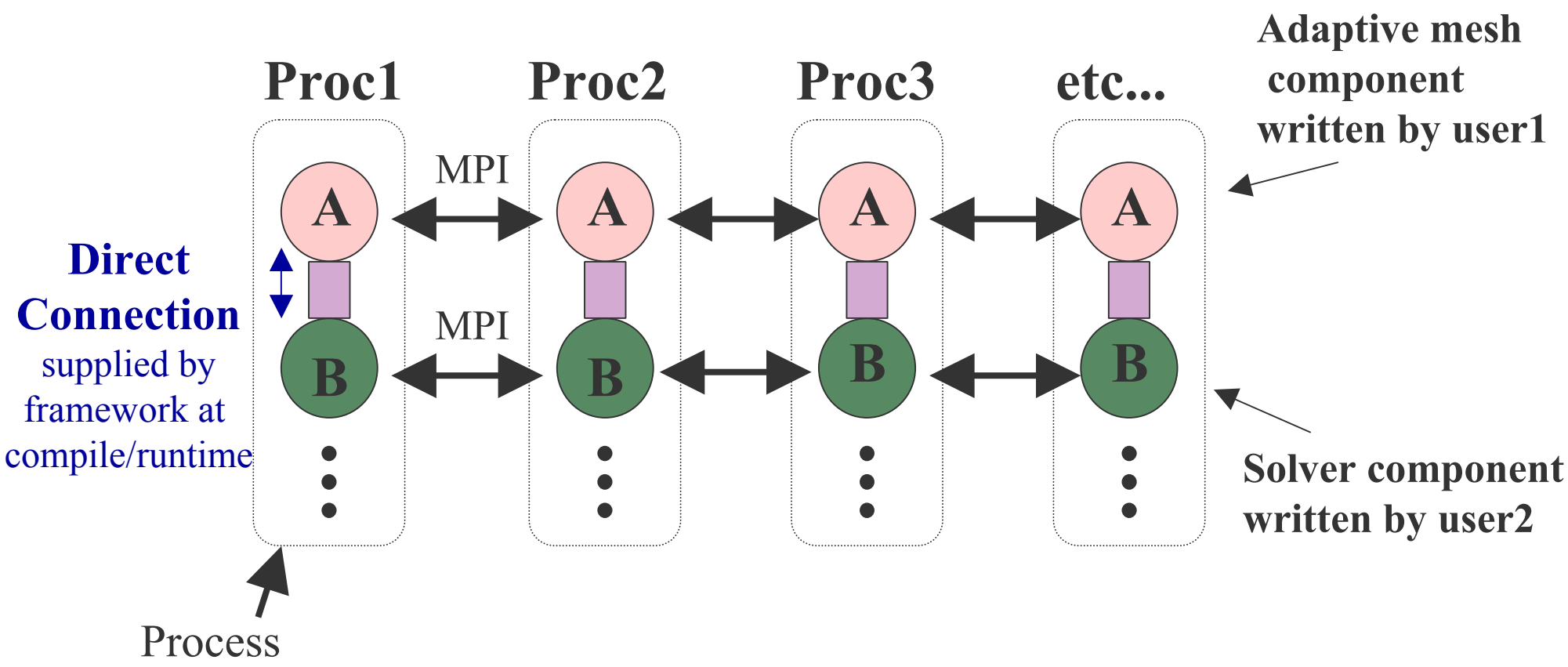
Software integration

- TOPS software achieves integration by supporting multiple interfaces
- Initially, this *N*-to-*N* compatibility is an $O(N^2)$ problem, dealt with case-by-case
- Once software is componentized and respects a standard interface, *N*-to-*N* compatibility reduces to an *O(N)* problem
- Overhead cost depends upon how deep into inner loops component interfaces occur; experience shows that significant interoperability costs only 1-5% overhead
- Reduces risk to applications developer, since all solvers are available
- Parallel generalization is "SCMD" (single-component, multiple data)



Schematic of SCMD components

MPI application using CCA for interaction between components A and B within the same address space

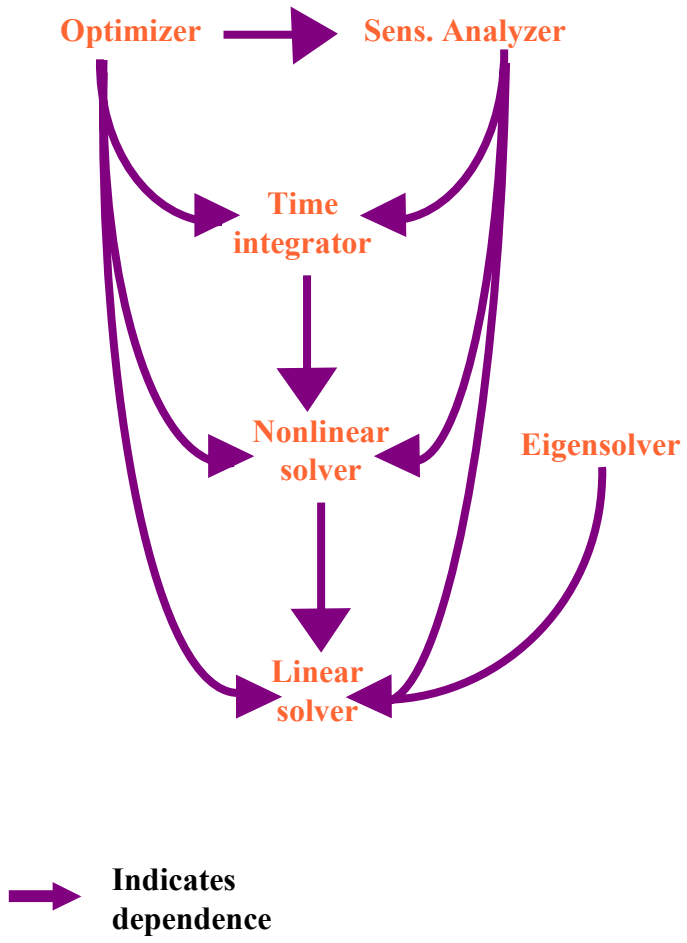


Software integration progress

- **Hypre in PETSc**
  - codes with **PETSc** interface (like CEMM's **M3D**) can invoke **Hypre** routines as solvers or preconditioners with command-line switch
- **SuperLU_DIST and Parallel_IC in PETSc**
  - invokable as above
- **Hypre in Chombo**
  - so far, **Hypre** is level-solver only; also FAC is being developed for AMR uses, like **Chombo**
- **Hypre and PETSc both being "SIDL'ized"**
  - one of TOPS' three foci of interaction with CCTTSS
- **TAO and PETSc componentized in early demonstration of CCA**
  - **DOE "Top 10" award** in 2002 recognized this effort, as part of larger componentization context

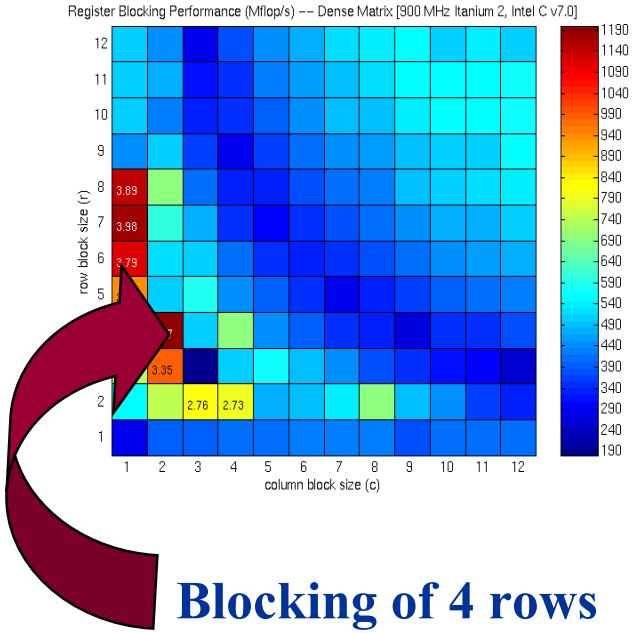
Performance optimization

- Optimal algorithms for large sparse matrices are prone to poor per-processor percentage of peak, since memory latency is ~100X processor clock period
- Critical to block sparse computations for registers and for cache
- TOPS leverages expertise that tuned dense kernels previously (ATLAS, PhiPAC)
- In 1999 TOPS researchers demonstrated gains of 2.5 to 7X over range of commercial microprocessors for NASA unstructured Euler code, from blocking and reordering (part of Bell Prize that year)
- Current efforts include atomic composite operations, common in solvers, e.g., $A^T Ax$



Performance optimization progress

- **TOPS has tuned sparse kernels**
  - (Jacobian) matrix-vector multiplication
  - sparse factorization
  - multigrid relaxation
- **Running on dozens of apps/platform combinations**
  - Power3 (NERSC) and Power4 (ORNL)
  - factors of 2 on structured (CMRS) and unstructured (CEMM) fusion apps
- **"Best student paper" at ICS2002 went to TOPS team**
  - theoretical model and experiments on effects of register blocking for sparse mat-vec



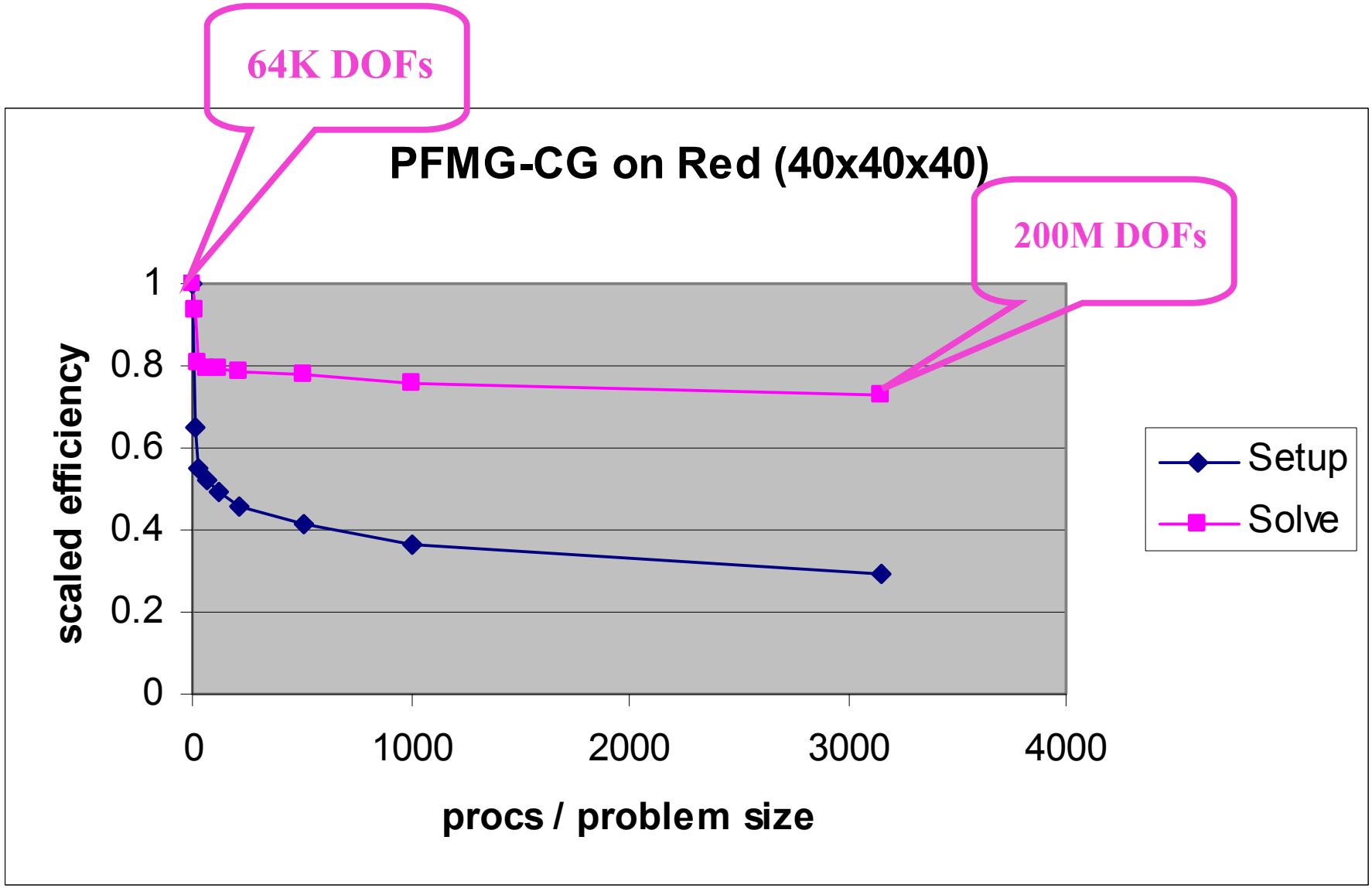
Blocking of 4 rows by 2 columns is 4.07 times faster on Itanium2 than default 1×1 blocks

SciDAC platforms

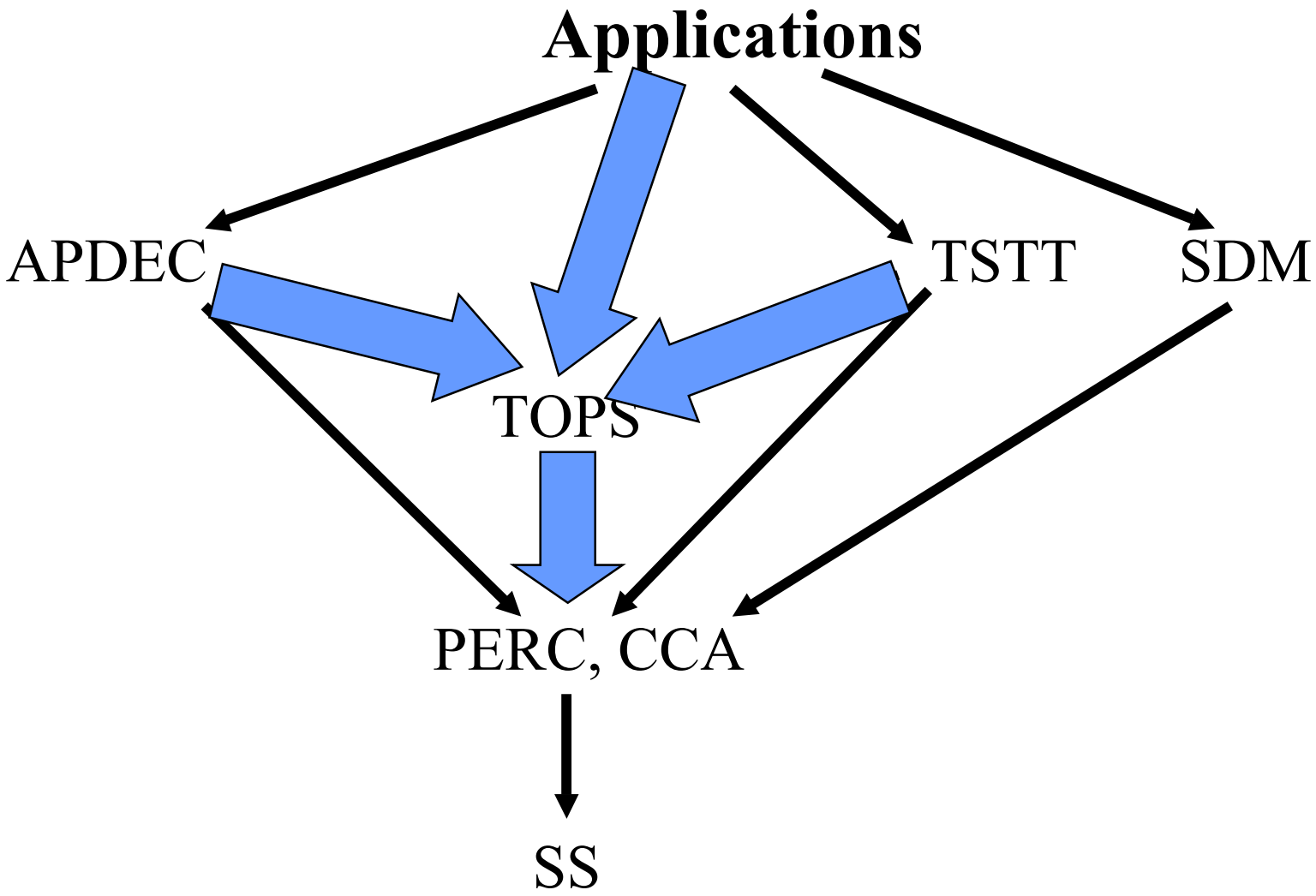
- IBM Power3+ SMP
- 16 procs per node
- 416 nodes
- 24 Gflop/s per node
- 10 Tflop/s

- IBM Power4 Regatta
- 32 procs per node
- 24 nodes
- 166 Gflop/s per node
- 4.5 Tflop/s

Parallel efficiency less a concern than serial!

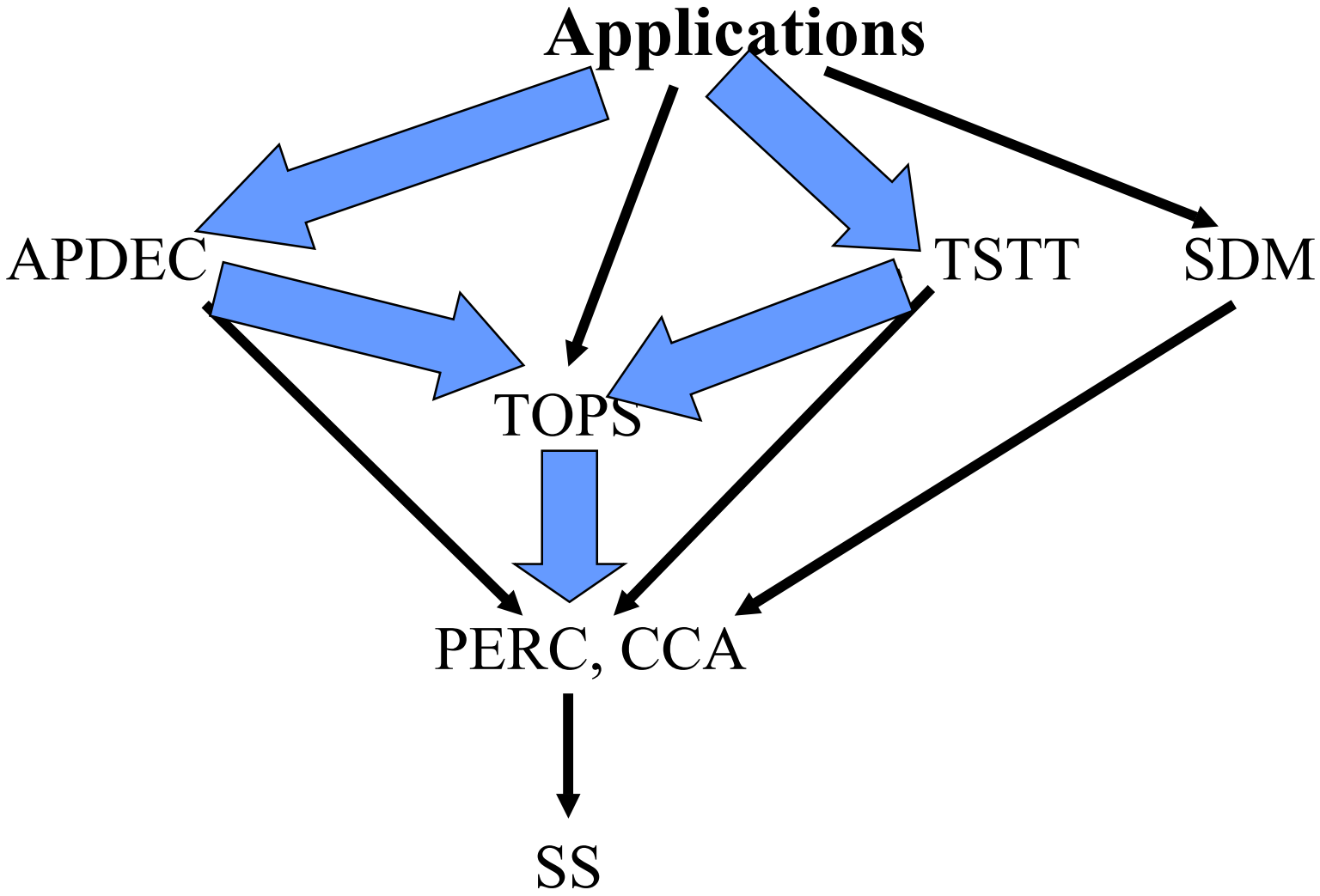


Primary interaction pathways, 2003*



*perspective of TOPS, not of our sponsors

Primary interaction pathways, 2005*



*perspective of TOPS, not of our sponsors

Lessons to date

- Working with the same code on the same machine vastly speeds collaboration, as opposed to ftp'ing matrices around the country, etc.
- Exchanging codes better than exchanging papers
- Version control systems essential to having any last impact or "insertion path" for solver improvements
- "Doing physics" more fun than doing driven cavities

TOPS software outreach

- **Hypre – scalable preconditioners**
  www.llnl.gov/CASC/hypre
- **PARPACK – scalable eigensolvers**
  hpcf.nersc.gov/software/libs/math/parpack
- **PETSc – scalable nonlinear and linear solvers**
  www.mcs.anl.gov/petsc
- **SUNDIALS – scalable ODE and nonlinear solvers**
  www.llnl.gov/CASC/sundials
- **SuperLU – parallel direct sparse LU methods**
  www.nersc.gov/~xiaoye/SuperLU
- **TAO – scalable general-purpose optimizers**
  www.mcs.anl.gov/tao
- **Veltisto – scalable PDE-constrained optimizers**
  www.cs.nyu.edu/~biros/veltisto

Seven questions for users

Has your solver been unchanged for the past five or ten years?

Is your solver running at 1-10% of machine peak?

Do you spend more time in your solver than in your physics?

Is your discretization or model fidelity limited by the solver?

Is your time stepping limited by stability?

Are you running loops *around* your analysis code?

Do you care how sensitive to parameters your results are?

If the answer to any of these questions is "yes", you may be a customer!

Expectations of users

- **Be willing to experiment with novel algorithmic choices – optimality is *rarely* achieved beyond model problems without interplay between physics and algorithmics!**
- **Adopt flexible, extensible programming styles in which algorithmic and data structures are not hardwired**
- **Be willing to let us play with the real code you care about, but be willing, as well to abstract out relevant compact tests**
- **Be willing to make concrete requests, to understand that requests must be prioritized, and to work with us in addressing the high priority requests**
- **If possible, *profile* before seeking help**

What we believe

- Many of us in TOPS came to work on solvers through interests in applications
- What we believe about …
  - applications
  - users
  - solvers
  - legacy codes
  - software

  … will impact how comfortable applications groups are collaborating with us

What we believe about *apps*

- **Solution of a system of PDEs is rarely a goal in itself**
  - **Actual goal is characterization of a response surface or a design or control strategy**
  - **Solving the PDE is just one forward map in this process**
  - **Together with analysis, sensitivities and stability are often desired**

⇒ **Software tools for PDE solution should also support related follow-on desires**

- **No general purpose PDE solver can anticipate all needs**
  - **Why we have *national laboratories*, not *numerical libraries* for PDEs today**
  - **A PDE solver improves with user interaction**
  - **Pace of algorithmic development is very rapid**

⇒ **Extensibility is important**

What we believe about *users*

- **Solvers are used by people of varying numerical backgrounds**
  - **Some expect MATLAB-like defaults**
  - **Others want to control everything, e.g., even varying the type of smoother and number of smoothings on different levels of a multigrid algorithm**

⇒ **Multilayered software design is important**

- **Users' demand for resolution is virtually insatiable**
  - **Relieving resolution requirements with modeling (e.g., turbulence closures, homogenization) only defers the demand for resolution to the next level**
  - **Validating such models requires high resolution**

⇒ **Processor scalability and algorithmic scalability (optimality) are critical**

What we believe about *legacy code*

- **Porting to a scalable framework does not mean starting from scratch**
  - **High-value meshing and physics routines in original languages can be substantially preserved**
  - **Partitioning, reordering and mapping onto distributed data structures (that we may provide) adds code but little runtime**

⇒ **Distributions should include code samples exemplifying "separation of concerns"**

- **Legacy solvers may be limiting resolution, accuracy, and generality of modeling overall**
  - **Replacing the solver may "solve" several other issues**
  - **However, pieces of the legacy solver may have value as part of a preconditioner**

⇒ **Solver toolkits should include "shells" for callbacks to high value legacy routines**

What we believe about *solvers*

- **Solvers are employed as part of a larger code**
  - **Solver library is not only library to be linked**
  - **Solvers may be called in multiple, nested places**
  - **Solvers typically make callbacks**
  - **Solvers should be swappable**

⇒ **Solver threads must not interfere with other component threads, including other active instances of themselves**

- **Solvers are employed in many ways over the life cycle of an applications code**
  - **During development and upgrading, robustness (of the solver) and verbose diagnostics are important**
  - **During production, solvers are streamlined for performance**

⇒ **Tunability is important**

What we believe about *software*

- A continuous operator may appear in a discrete code in many different instances
  - Optimal algorithms tend to be hierarchical and nested iterative
  - Processor-scalable algorithms tend to be domain-decomposed and concurrent iterative
  - Majority of progress towards desired highly resolved, high fidelity result occurs through cost-effective low resolution, low fidelity parallel efficient stages

⇒ Operator abstractions and recurrence are important

- Hardware changes many times over the life cycle of a software package
  - Processors, memory, and networks evolve annually
  - Machines are replaced every 3-5 years at major DOE centers
  - Codes persist for decades

⇒ Portability is critical

"There will be opened a gateway and a road to a large and excellent science into which minds more piercing than mine shall penetrate to recesses still deeper." – **Galileo (1564-1642), on *'experimental mathematics'* appropriated here for *'simulation science'***

http://www.tops-scidac.org